Pattern Recognition 65 (2017) 11–25

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Variational method for joint optical flow estimation and edge-aware image restoration

Zhigang Tu[a,*], Wei Xie[b], Jun Cao[c], Coert van Gemeren[a], Ronald Poppe[a], Remco C. Veltkamp[a]

[a] Department of Information and Computing Sciences, Utrecht University, Princetonplein 5, Utrecht, The Netherlands
[b] School of Computer, Central China Normal University, Luoyu Road 152, Wuhan, China
[c] Intel Corp., 4500 S. Dobson Road, Chandler, AZ 85224, USA

ARTICLE INFO

Keywords:
Optical flow
Image sequence restoration
Edge preserving
Efficient numerical solver

ABSTRACT

The most popular optical flow algorithms rely on optimizing the energy function that integrates a data term and a smoothness term. In contrast to this traditional framework, we derive a new objective function that couples optical flow estimation and image restoration. Our method is inspired by the recent successes of edge-aware constraints (EAC) in preserving edges in general gradient domain image filtering. By incorporating an EAC image fidelity term (IFT) in the conventional variational model, the new energy function can simultaneously estimate optical flow and restore images with preserved edges, in a bidirectional manner. For the energy minimization, we rewrite the EAC into gradient form and optimize the IFT with Euler–Lagrange equations. We can thus apply the image restoration by analytically solving a system of linear equations. Our EAC-combined IFT is easy to implement and can be seamlessly integrated into various optical flow functions suggested in literature. Extensive experiments on public optical flow benchmarks demonstrate that our method outperforms the current state-of-the-art in optical flow estimation and image restoration.

1. Introduction

Optical flow estimation aims at calculating a pixel-wise displacement field between two images. It is the most fundamental and well-investigated problems in computer vision [1]. The flow field provides crucial correspondence information that can be used in a variety of visual tasks such as image interpolation [2], super-resolution reconstruction [3], object segmentation and tracking [4,5], action recognition [6,7] and autonomous navigation [8,9].

Variational method, which is the current predominant way to estimate dense optical flow [10], has been proposed by Horn and Schnuck (HS) [11]. It combines a brightness constancy assumption (BCA)-based data term with a global smoothness term. The data term assumes that the brightness of corresponding pixels does not change during motion. This single constraint is insufficient to determine two unknown components of the optical flow (see Eq. (3)), which is known as the aperture problem. To handle the aperture problem, HS introduced as a smoothness term, which is based on the assumption that the flow vector varies smoothly over the flow field. In practice, however, these two constraints are often violated, which prevents the variational method to robustly treat image noise, handle illumination changes and large displacements, and to preserve flow discontinuities.

Various extensions and improvements have been proposed to handle these problems [10]. In general, these variations can be classified into three categories [12]: pre-processing of the input images, modification of the variational formulation and post-processing of the flow field.

Pre-processing is often applied to remove image artifacts of the input images, including noise filtering [13,14] and deblurring [15].

Modifications are the second category of variations of the traditional formulation. They are aimed at improving the data term to make the algorithm more resistant to noise [16,17], more robust under illumination changes [18–20], and more capable to deal with large displacements [10,1,21]. In addition, refining the smoothness term to preserve motion discontinuity [22].

One challenge for optical flow computation is to deal with large displacements. Recently, there has been an interest in using feature matching to assist the variational methods to address this problem. By matching image features, reliable correspondence information between two images can be obtained [21]. The feature matching and variational methods are complementary, but combining them is not straightforward. Feature matching exploits a discrete space of admissible correspondences, whereas the variational model imposes linearization during the differential optimization procedure [23]. Brox and Malik

* Corresponding author.
E-mail addresses: Zhigang.Tu@asu.edu (Z. Tu), XW@mail.ccnu.edu.cn (W. Xie), Jun.Cao@intel.com (J. Cao), C.J.Vangemeren@uu.nl (C. van Gemeren), R.W.Poppe@uu.nl (R. Poppe), R.C.Veltkamp@uu.nl (R.C. Veltkamp).

http://dx.doi.org/10.1016/j.patcog.2016.10.027
Received 15 April 2016; Received in revised form 20 October 2016; Accepted 22 October 2016
Available online 24 October 2016
0031-3203/ © 2016 Elsevier Ltd. All rights reserved.

[10] add a feature matching constraint into the variational framework. In this way, correspondences from descriptor matching are obtained, which are helpful to support the coarse-to-fine/warping strategy in avoiding local minima. But the local descriptors are reliable only at salient locations, and false matches are common. Chen et al. [21] used approximate nearest neighbor fields (NNF) to compute an initial motion field that contains different types of correspondence information, and then fuse it with variational flow results for refinement.

Selecting an appropriate weighting parameter λ to obtain a better balance between the data term and the smoothness term [24,25] is a way to improve the performance of optical flow estimation. Weight selection based on the weighted distance using the blurring operator is studied in [26], but does not directly apply to variational algorithms because of its spatially varying character. Zimmer et al. [22] proposed an optimal prediction principle (OPP) to automatically determine the optimal smoothness parameter according to an average data constancy error (ADCE) measure. The performance of the OPP method is limited to conditions with constant speed and linear trajectories of objects. Raket [27] presented a local data fidelity method to fuse flows of different smoothness parameters. However, the local evaluation in terms of the best data fit on the gradient images is easily affected by illumination changes and it is sensitive to noise and edge discontinuities. Tu et al. [24] improved the method of [27] by evaluating the local interpolation error in terms of weighted L1 block matching on the corresponding set of intensity images.

A third category of improvements to the variational method is post-processing of the flow field, such as smoothing the flow field to remove motion inaccuracies [28,29].

Most of these variations are closely related to the original HS framework, which depends on a data term and a smoothness term. Few works introduced additional terms to reconstruct a more effective optical flow formulation. The majority of existing variational optical flow methods treat the input images in the image-based data term and the flow in the flow-based smoothness term separately during iterative optimization [30]. But they are correlated and affect each other. The smoothness term measures variation of the flow field, which relies on the estimated flow itself. The data term measures deviations from the optical flow equation, where the deviations originate from both the inaccuracies of the flow field and the artifacts of images caused, amongst others, by noise and motion blur. Image artifacts and inaccuracies in flow field should both be handled since the energy function is a weighted sum of them. We thus need to simultaneously denoise the images and refine the intermediate flow fields.

In this paper, we propose a new optical flow energy function that integrates an edge-aware constraints (EAC) added image fidelity term (IFT) [30] to simultaneously estimate optical flow and restore image within a variational framework. We make the following contributions:

- We introduce an additional IFT, which penalizes deviations from the measured image. This term aims to restore the input images by denoising. To better preserve edges during filtering, we explicitly add EAC [31]. The constraints enforce similar image filtering effects to be propagated in local areas with similar appearance to preserve edges.
- To minimize the EAC energy function, we rewrite the EAC in the form of first-order gradient constraints. Then, the input images are efficiently restored with edges preserved by optimizing the corresponding Euler–Lagrange equations of the integrated IFT. The linear system of equations can be solved analytically directly (this paper) or by numerical methods (e.g. successive over-relaxation, SOR).

The remaining paper is organized as follows. We first review related works on dense optical flow estimation and edge-preserving denoising. Section 3 describes our approach. The optimization framework is introduced in Section 4. We describe the implementation in Section 5. We evaluate our method qualitatively and quantitatively in Section 6. Finally, discussion and conclusions are given in Section 7.

2. Related work

There is a large body of work on optical flow following the variational model of HS [11]. It is beyond the scope of this paper to review all works. Instead, we only discuss the most relevant papers, the work that addresses simultaneously solving the optical flow and filtering the image, and image denoising is carried out on gradient domain with edge-preserving.

Optical flow computation with image sequence restoration. Solving the optical flow computation and image sequence restoration in a unified framework has rarely been studied. Still, a simultaneous approach is beneficial for improving the accuracy of the estimated flow field, which is especially useful for noisy image sequences. In traditional optical flow algorithms, the flow computation is independent of the image restoration. For optical flow estimation, images are typically pre-filtered to remove noise [13,14], to reduce effect of shadows [32] or to handle illumination changes [33]. Some methods concentrate on filtering techniques to smooth the intermediate flow field by removing outliers or correcting flow errors [12]. Xiao et al. [34] and Drulea and Nedevschi [35] applied the bilateral filter to smooth the flow field with desirable motion boundary preservation. Xu et al. [1] used a bilateral filter to detect occluded pixels and refine the flow to handle occlusions. Wedel et al. [14] introduced a median filter (MF) to remove noise from the flow field, but the MF over-smoothes the flow field edges. Sun et al. [36] proposed a modified weighted median filter (WMF) which depends on the spatial and color distance between pixels to prevent this kind of over-smoothing.

For image restoration, some methods just use the pre-computed flow as auxiliary information. For example, [37] used optical flow to guide the temporal derivative during restoration. In the mentioned methods, the flow computation is not interacting with the image restoration. But these methods indicate that the estimated flow and the restored image are related and can benefit each other.

Wulff and Black [38] applied a layered model to jointly estimate motion and deblurring layers. The most relevant work to tackle this problem is Nir et al. [30], which jointly computes optical flow and denoises the image sequence within a unified variational framework. Two main contributions are made in their work. First, an additional IFT is introduced to construct a new energy function. The IFT is crucial for keeping the restored images close to the input images. Second, they optimize the new energy function by alternately minimizing an optical flow energy function and a denoising restoration function. Optical flow can be computed in terms of numerical optimization methods [18], and the images can be restored using a gradient descent technique. However, the IFT-based denoising scheme cannot preserve edges well. Additionally, the gradient descent optimization method is computationally expensive and often falls into local minimum. To address these problems, we integrate EAC to the IFT to better preserve edges and to save computation time.

Edge-preserving denoising. Several edge-preserving filtering approaches have been proposed to avoid artifacts in image denoising. The total variation-based image restoration model was first introduced by Rudin–Osher–Fatemi (ROF [39]) to preserve edges during image denoising. It is designed with the explicit goal of preserving sharp edges in the image while removing noise and other unwanted artifacts. However, the ROF model also penalizes large gradient magnitudes that possibly affect contrast during denoising. To handle this problem, [40] presented a new image filtering method for sharpening major edges by increasing the steepness of transition while eliminating low-amplitude structures. They achieved the contradictive effect by making use of L0 gradient minimization to globally control how many non-zero gradients were obtained to approximate prominent structures. The L0 gradient minimization strategy of [40] is not suitable for image denoising within

the variational framework as it is based on discrete counting manner. Recently, inspired by the edit propagation, Hua et al. [31] proposed EAC to better preserve edges for general gradient domain image filtering. More importantly, the EAC can be seamlessly integrated with the IFT to increase its performance on preserving edges of images, as we demonstrate in this paper.

3. Variational method for joint optical flow estimation and edge-aware image restoration

Traditional optical flow algorithms [16,13,14] typically pre-filter the input images to reduce noise, an operation that does not rely on the estimated flow. For image restoration, most techniques calculate the optical flow as a preprocessing step independent of the restoration processing. In this section, a variational model for joint optical flow estimation and edge-aware image restoration is introduced. Due to the introduced EAC, images can be restored by minimizing the corresponding Euler–Lagrange equations of the newly EAC added IFT analytically, which is more efficient and accurate than the gradient descent method of [30].

3.1. Classical variational optical flow algorithm

In the classical work, Nir et al. [30] stated that there are two sources for errors in the variational optical flow data term: (1) inaccuracy in the flow field and (2) artifacts in the input sequence due to noise, motion blur, etc. Based on this characteristic, they proposed a new energy function to compute optical flow and restore images jointly:

$$E(u, v, I) = E_{\mathrm{D}}(u, v, I) + \lambda E_{\mathrm{S}}(u, v) + \alpha E_{\mathrm{F}}(I, f) \tag{1}$$

where λ is the smoothness parameter which determines the tradeoff between the data term E_{D} and smoothness term E_{S}. $I = (I_1, I_2)$ represents the two consecutively restored frames, and $f = (f_1, f_2)$ denotes the two consecutively original frames. (u, v) is the flow field, with u the horizontal and v the vertical flow component. The direction and magnitude of each flow component in flow field indicates where and how a pixel moved between I_1 and I_2. α is for tradeoff between the optical flow term and the restoration term.

In general, the artifacts that are caused by noise in the input images can be modeled as [30]:

$$f = I + n \tag{2}$$

where n represents the additive noise.

The variational framework models errors more comprehensively than traditional algorithms [16,11,18], since it specifies errors on both the estimated flow field and the restored and warped interpolation image. For minimization, different from most of the traditional flow energy functions which only depend on the optical flow (u, v), this new energy function considers both the estimated optical flow (u, v) and the restored images (I_1, I_2). Consequently, optical flow and image denoising can be interactively solved. More importantly, during the warping-based optimization, the (u, v) and the (I_1, I_2) refine each other after every iteration.

$E_{\mathrm{D}}(u, v, I)$ is the data term, based on the BCA, and has the form:

$$E_{\mathrm{D}}(u, v, I) = \int_{\Omega} \Psi_{\mathrm{D}}((I_2(x + u, y + v) - I_1(x, y))^2) dxdy \tag{3}$$

where Ψ is a penalty function, and x=(x, y) denotes a point in the image domain Ω. As suggested in [28], we select the generalized Charbonnier robust penalty function $\Psi_{\mathrm{D}}(\mathbf{x}^2) = (\mathbf{x}^2 + \xi^2)^{\beta}$ with settings $\xi = 0.001$ and $\beta = 0.45$ for the data term.

$E_{\mathrm{S}}(u, v)$ is the smoothness term that quantifies the smoothness of the flow field:

$$E_{\mathrm{S}}(u, v) = \int_{\Omega} \Psi_{\mathrm{S}}(\|\nabla u\|^2 + \|\nabla v\|^2) dxdy \tag{4}$$

where $\Psi_{\mathrm{S}}(\mathbf{x}^2) = (\mathbf{x}^2 + \xi^2)^{\beta}$. The robust smoothness term helps the data term to solve the aperture problem, and it is also useful to deal with outliers.

The additional $E_{\mathrm{F}}(I, f)$ refers to IFT, which penalizes deviations from the noisy input images [30]. It is crucial to keep $I = (I_1, I_2)$ close to $f = (f_1, f_2)$. $E_{\mathrm{F}}(I, f)$ is defined as:

$$E_{\mathrm{F}}(I, f) = \int_{\Omega} (I - f)^2 dxdy \tag{5}$$

Due to the IFT, this new technique obtains more accurate flow results compared to the previous algorithms without coupled image denoising [30]. Therefore, we introduce an effective EAC [31] to incorporate with the IFT to deal with this problem.

3.2. Edge-aware constraints

In classical image filtering work, the objective function is formed as squared errors between the restored image and the input image. No constraints are included to prevent image filtering to propagate to neighboring similar objects. To better preserve edges for general gradient domain image filtering, [31] introduced EAC to these methods. The constraints are expressed as:

$$E_{\mathrm{E}}(I, f) = \sum_{\mathrm{x}' \in N_4(\mathrm{x})} w(\mathrm{x}, \mathrm{x}')[(I(\mathrm{x}) - f(\mathrm{x})) - (I(\mathrm{x}') - f(\mathrm{x}'))]^2 \tag{6}$$

where $N_4(\mathrm{x})$ is the set of 4-connected neighbors (i.e. x′) of pixel x. $I(\mathrm{x}) - f(\mathrm{x})$ measures the changes by image filtering locally. $w(\mathrm{x}, \mathrm{x}')$ are color similarity weights defined as Gaussian differences between adjacent color vectors $\mathbf{I}(x, y)$ and $\mathbf{I}(x', y')$ in the CIELab space as in [28,29]:

$$w(\mathrm{x}, \mathrm{x}') = \exp\left(-\frac{\|\mathbf{I}(x, y) - \mathbf{I}(x', y')\|^2}{2\sigma_I^2}\right) \tag{7}$$

This weight function is effective and gives more weight to pixels located in the same surface.

The EAC avoid averaging smoothing which is typical for local methods [41], and enforce similar image smoothing effects to be propagated to areas with similar appearance. Consequently, the EAC prevent artifacts such as gradient reversals at image edges, and improve the performance on preserving edges of normal gradient domain image filtering methods.

More importantly, the EAC can be incorporated into the general gradient domain optimization framework seamlessly in the form of first-order gradient:

$$E_{\mathrm{E}}(I, f) = \sum_{\mathrm{x}' \in N_4(\mathrm{x})} w(\mathrm{x}, \mathrm{x}')[(I(\mathrm{x}) - I(\mathrm{x}')) - (f(\mathrm{x}) - f(\mathrm{x}'))]^2 \tag{8}$$

As x′ represents the set of 4-connected neighbors of x, Eq. (8) is actually the sum of 4 spatial discrete partial derivatives at each point x:

$$\begin{aligned} E_{\mathrm{E}}(I, f) = {} & w_{x+}[-(I(\mathrm{x})_{x+} - f(\mathrm{x})_{x+})]^2 + w_{x-}[I(\mathrm{x})_{x-} - f(\mathrm{x})_{x-}]^2 \\ & + w_{y+}[-(I(\mathrm{x})_{y+} - f(\mathrm{x})_{y+})]^2 + w_{y-}[I(\mathrm{x})_{y-} - f(\mathrm{x})_{y-}]^2 \end{aligned} \tag{9}$$

where $x+$ and $x-$ denote the forward and backward difference along x-direction respectively. In particular, $I(\mathrm{x})_{x+} = -(I(x, y) - I(x + 1, y))$ and $I(\mathrm{x})_{x-} = I(x, y) - I(x - 1, y)$. Other components are treated in the same numerical manner.

In this work, inspired by the advantages of the EAC, we combine them into the variational model to jointly improve the performances of optical flow estimation and image restoration.

3.3. Edge-aware variational optical flow algorithm

In the baseline method [30], the IFT $(E_{\mathrm{F}}(I, f))$ is effective in restoring input images by denoising them. However, it does not preserve edges because no first-order constraints are added to obtain

good pixel-gradients. To handle this issue, we integrate Eq. (8) into Eq. (1) to form an edge-aware variational model:

$$E(u, v, I) = E_D(u, v, I) + \lambda E_S(u, v) + \alpha E_F(I, f) + \gamma E_E(I, f) \tag{10}$$

where γ is used to control the balance between the EAC and the original 0th-order IFT.

To compute the spatial derivatives of the images more efficiently and to minimize the new energy function more easily, we replace the independent forward and backward difference with a unified derivative by using a difference filter $[-1\ 0\ 1]/2$. The x-direction derivative of I is calculated as:

$$I(\mathrm{x})_x = [((I(x+1, y) - I(x, y)) + (I(x, y) - I(x-1, y))] /2 = [-1\ \ 0\ \ 1]/2 * I(\mathrm{x}) \tag{11}$$

where * denotes convolution. The color similarity weight $w(\mathrm{x}, \mathrm{x}')$ is recomputed as:

$$w_x(\mathrm{x}, \mathrm{x}') = \exp\left(-\frac{|\mathbf{I}(\mathbf{x})_x|^2}{2\sigma_I^2}\right) w_y(\mathrm{x}, \mathrm{x}') = \exp\left(-\frac{|\mathbf{I}(\mathbf{x})_y|^2}{2\sigma_I^2}\right) \tag{12}$$

The added EAC in the form of first-order gradients are expressed as:

$$E_E(I, f) = w_x|I_x - f_x|^2 + w_y|I_y - f_y|^2 \tag{13}$$

Finally, our edge-aware variational energy function is defined as:

$$\begin{aligned} E(u, v, I) = \lambda & \int_\Omega \Psi_S(\|\nabla u\|^2 + \|\nabla v\|^2) dxdy \\ & + \int_\Omega \Psi_D((I_2(x+u, y+v) - I_1(x, y))^2) dxdy \\ & + \int_\Omega [\alpha((I_1 - f_1)^2 + (I_2 - f_2)^2) + \gamma(w_{1x}|I_{1x} - f_{1x}|^2 \\ & + w_{1y}|I_{1y} - f_{1y}|^2 + w_{2x}|I_{2x} - f_{2x}|^2 + w_{2y}|I_{2y} - f_{2y}|^2)] dxdy \end{aligned} \tag{14}$$

The third term is a modified ROF model with two constraints: the 0th-order components (i.e. the image fidelity) imposed to get desired pixel values, and the first-order components (i.e. the EAC) imposed to get desired pixel-gradients over space and time.

4. Optimization framework

Our new objective model is a function of variables (u, v, I_1, I_2). To simultaneously solve all the unknowns, an alternating minimization framework [28] can be used. Since the optical flow terms (i.e. the first and second terms of Eq. (14)) depend on the restored images, and the image restoration terms (i.e. the second and third terms of Eq. (14)) depend on the computed optical flow, we split the objective function into two coupled modules:

$$\begin{aligned} E_O(u, v, I) = \lambda & \int_\Omega \Psi_S(\|\nabla u\|^2 + \|\nabla v\|^2) dxdy \\ & + \int_\Omega \Psi_D((I_2(x+u, y+v) - I_1(x, y))^2) dxdy \end{aligned} \tag{15}$$

and

$$\begin{aligned} E_R(u, v, I) = & \int_\Omega \Psi_D((I_2(x+u, y+v) - I_1(x, y))^2) dxdy \\ & + \int_\Omega [\alpha((I_1 - f_1)^2 + (I_2 - f_2)^2) + \gamma(w_{1x}|I_{1x} - f_{1x}|^2 + w_{1y} \\ & |I_{1y} - f_{1y}|^2 + w_{2x}|I_{2x} - f_{2x}|^2 + w_{2y}|I_{2y} - f_{2y}|^2)] dxdy \end{aligned} \tag{16}$$

The alternating optimization strategy first holds I_1 and I_2 fixed, and then minimizes the optical flow module equation (15) by solving its Euler–Lagrange equations with respect to variables u and v:

$$\begin{aligned} \Psi'(I_t^2) I_t I_x - \lambda \operatorname{div}(\Psi'(\|\nabla u\|^2 + \|\nabla v\|^2)\nabla u) = 0 \Psi'(I_t^2) I_t I_y \\ - \lambda \operatorname{div}(\Psi'(\|\nabla u\|^2 + \|\nabla v\|^2)\nabla v) = 0 \end{aligned} \tag{17}$$

where

$$\begin{aligned} I_t &= I_2(x+u, y+v) - I_1(x, y); I_x = \partial_x I_2(x+u, y+v); I_y \\ &= \partial_y I_2(x+u, y+v). \end{aligned} \tag{18}$$

The two nested fixed-point iteration numerical scheme of [18] is used to solve the flow (u, v).

In the second step, with flow (u, v) fixed, we minimize the image denoising module equation (16) to restore I_1 and I_2. Note that Eq. (16) is a formulation of $E(x, y, I_1, I_2, I_{1x}, I_{1y}, I_{2x}, I_{2y})$. It can be solved using Euler–Lagrange equations with respect to variables I_1 and I_2:

$$\begin{aligned} & \alpha I_1 - \alpha f_1 - \Psi'(I_t^2) I_t - \gamma\left(\frac{w_{1x}|I_{1x} - f_{1x}|}{\partial x} + \frac{w_{1y}|I_{1y} - f_{1y}|}{\partial y}\right) \\ & = 0 \alpha I_2 - \alpha f_2 - \gamma\left(\frac{w_{2x}|I_{2x} - f_{2x}|}{\partial x} + \frac{w_{2y}|I_{2y} - f_{2y}|}{\partial y}\right) = 0 \end{aligned} \tag{19}$$

The derivatives, for example $\frac{|I_{1x} - f_{1x}|}{\partial x}$ and $\frac{|I_{1y} - f_{1y}|}{\partial y}$, are approximated with a 5-point stencil $\frac{1}{12}[-1\ 8\ 0\ -8\ 1]$ as in [14]. The linear equation system (19) can be solved either by common numerical methods (e.g. SOR [42]) to get the arithmetical solution or by direct computation to get the analytical solution as follows:

$$\begin{aligned} I_1 &= f_1 + \frac{\gamma}{\alpha}\left(\frac{w_{1x}|I_{1x} - f_{1x}|}{\partial x} + \frac{w_{1y}|I_{1y} - f_{1y}|}{\partial y}\right) + \frac{1}{\alpha}\Psi'(I_t^2) I_t I_2 \\ &= f_2 + \frac{\gamma}{\alpha}\left(\frac{w_{2x}|I_{2x} - f_{2x}|}{\partial x} + \frac{w_{2y}|I_{2y} - f_{2y}|}{\partial y}\right) \end{aligned} \tag{20}$$

In this coarse-to-fine optimization framework, on each pyramid, these two modules are alternatingly optimized at every iteration to refine the solution. The restored images are supplied as initialization for the optical flow module in the next iteration.

Since γ controls the balance between the EAC and the original image restoration term, it should be determined appropriately. We discuss this drawback in the last section. We empirically find that $\gamma = 1$ and $\alpha = 1$ are suitable values.

4.1. Edge-aware constraints added image fidelity term vs. image fidelity term

If we do not add the EAC to the IFT, the IFT-based image restoration module of [30] is expressed as:

$$\begin{aligned} E(u, v, I) = & \int_\Omega \Psi_D((I_2(x+u, y+v) - I_1(x, y))^2) dxdy \\ & + \alpha \int_\Omega ((I_1 - f_1)^2 + (I_2 - f_2)^2) dxdy \end{aligned} \tag{21}$$

There are two differences between the IFT [30] (Eq. (21)) and our EAC added IFT (Eq. (16)).

First, as Hua et al. [31] stated, Eq. (21) specifies the squared error between the desired values and the actual values. However, there are no explicit constraints to restraint the image filter effect not to propagate to adjacent similar objects, so edges cannot be preserved well.

Second, the optimization method is different. Due to the introduced gradient components $(I_{1x}, I_{1y}, I_{2x}, I_{2y})$, the energy function equation (16) can obtain an extremum according to its corresponding Euler–Lagrange equations (i.e. Eq. (19)). In other words, the stationary value of the two unknowns I_1 and I_2 can be computed. However, it is hard for the non-convex energy function equation (21) to obtain an extremum. Furthermore, the performance of the gradient descent method that is used to solve Eq. (21) heavily depends on the initialization and the step size, but is hard to determine the proper initialization and step size. Besides, the gradient descent method is slow. In contrast, the added EAC allows for an efficient closed analytical solution using Eq. (20).

5. Implementation

To evaluate the effectiveness of our EAC technique in improving optical flow accuracy and preserving edges of the restored images, we compare it to the state-of-the-art optical flow algorithms [19–21,43,35,1,28,36].

Refs. [19,20,43,35] address the issue of illumination changes in optical flow estimation. The NLTGV-SC [19] introduces a higher-order regularization term to accurately localize motion boundaries and to resolve ambiguities in the matching term. A matching term is presented, robust to illumination and scale changes, to address the two sources of errors in optical flow computation. MLDP-OF [20] applies a texture constancy assumption to replace the traditional brightness constancy assumption to deal with illumination changes. ROF-NND [43] is multi-resolution TV-L1 method with a data-term based on neighborhood descriptors and a weighted non-local regularization term to increase the robustness to illumination changes. The Corr.Flow [35] (code available at http://cv.utcluj.ro/optical-flow.html) uses a zero-mean normalized cross-correlation matching measure to handle the issue of integrating matching functions into the variational framework, and performs well in the presence of illumination changes.

MDP-Flow [1] (code available at http://www.cse.cuhk.edu.hk/leojia/projects/flow/) is an extended coarse-to-fine framework to reduce the reliance of flow estimates on their initial values propagated from the coarse level. By integrating matching information into the flow at each scale, motion details can be recovered and large displacements can be addressed. Classic+NL [36] (code available at http://cs.brown.edu/ dqsun/code/cvpr10_flow_code.zip) is a comprehensive method that combines various modern optimization and implementation techniques within the classical flow formulations, and produces competitive results. For example, the structure–texture decomposition [36] technique is used to treat illumination changes. A WMF is proposed to reduce outliers. Classic+NLP [28] introduces an asymmetric pyramid downsampling scheme, which applies an unequal downsampling factor in each direction to ensure that the motion at the top pyramid level is small in both directions, to improve the performance of the Classic+NL in estimating longer range motions. NN-field [21] (or updated version NNF-Local [44]) (code available at https://sites.google.com/site/levchen2010/home/publications) is derived from [36]. By initializing the motion field with refined nearest neighbor fields (NNF), it ensures that the Classic+NL can handle large displacements. The results of NNF (NoRest.) in our experiments are computed by the code NN-field [21] supplied. Furthermore, we compare against the method of Portz et al. [45] that concerns spatially-varying motion blur (code available at http://pages.cs.wisc.edu/ lizhang/projects/blurflow/). Finally, the classical approach of Nir et al. [30], which is most related to our method, is selected for comparison.

Quantitative and visual evaluations are conducted on both synthetic and real sequences from public benchmarks: Middlebury [13], KITTI [46], BlurSequences [38], MIT [47] and MPI-Sintel [48]. Experiments are conducted on a laptop with an Intel Core i5-2410M 2.30 GHz processor and 4 GB memory. In our current CPU implementation, the whole program takes 380 s to compute a high quality flow field for an image pair with resolution 640×480 in, for instance, the *Urban* sequence. The pixel values of the experimental images are in the range [0, 255].

To be robust against illumination changes, we preprocess the input images by applying the ROF-based [39] structure–texture decomposition [36], where the images are reconstructed by linearly combine the texture and structure components in the proportion 20:1 as [14]. The incremental coarse-to-fine technique in combination with the fixed pointed iteration scheme [18] is used to estimate the optical flow. In the outer loop of the optimization, a graduated nonconvexity (GNC) strategy is used [49], which linearly combines a quadratic penalty function with a generalized Charbonnier penalty function [36]: $E_r(u, v) = (1-\kappa)E_Q(u, v) + \kappa E_C(u, v)$. Three GNC steps with $\kappa = \{0, 0.5, 1\}$ respectively are implemented. In building the pyramid, we use a downsampling factor of 0.5 as [36], and the number of pyramid levels is adaptively determined with a constraint that the coarsest level has a height or width larger than or equal to 20 pixels. In each warping iteration step, flow computation and image denoising restoration are jointly performed.

Table 1
The results are different when selecting different values of λ, the changes due to λ are not sensitive to the EAC added IFT part (on the Middlebury training set).

λ	$\lambda = 0.5$	$\lambda = 1.0$	$\lambda = 3.0$	$\lambda = 5.0$	$\lambda = 7.0$
Avg.AAE	2.622	2.481	**2.476**	2.546	2.649
Avg.AEE[1]	0.224	0.213	**0.204**	0.209	0.215

[1] The bold values is best.

As shown in Table 1 (Avg.AAE/Avg.AEE represent the average AAE/AEE of all 8 sequences, where AEE and AEE are defined in Eqs. (22) and (23) respectively), our EAC added IFT is not sensitive to λ. $\lambda = 3$, which is the suggested value in [36,28,12], also performs best in our model on the training sequences from the Middlebury dataset. Therefore, in this paper, we select $\lambda = 3$ for all the experiments. We provide the Matlab source code and the experimental results at http://www.projects.science.uu.nl/opticalflow/NNF-EAC/. The pseudo-code of the EAC integrated NN-field [21], referred to as NNF-EAC, is summarized in Algorithm 1.

Algorithm 1. NNF-EAC algorithm.

Input: Images f_1 and f_2
Output: Flow field (u,v) and restored images I_1 and I_2
Pre-process f_1 and f_2 (structure–texture decomposition [36])
Pre-compute a NNF $(\hat{u}, \hat{v})$ from f_1 and f_2
Pre-process f_1 and f_2 to get I_1 and I_2
for l=1 to *maxlevel* **do**
  Compute pyramid images ${}^{l}I_1$ and ${}^{l}I_2$
  Compute pyramid NN-field flow [21] $({}^{l}\hat{u}, {}^{l}\hat{v})$
end
for l=L to 1 (set L = *maxlevel*)
  /****Initialization****/
  (1)initialize the continuous flow before first warping iteration on each pyramid:
  if l=L, $({}^{L}u, {}^{L}v) = 0$; else, $({}^{l}u, {}^{l}v) = resample({}^{l+1}u, {}^{l+1}v)$
  (2)refine the initialized continuous flow with NN-field flow by fusion:
  $({}^{l}u, {}^{l}v) = \text{fusion}\{({}^{L}u, {}^{L}v), ({}^{L}\hat{u}, {}^{L}\hat{v})\}$
  /****Flow computation and Image restoration****/
  for k=1 to *warp_iteration*
    Flow computation using method [28]:
    Initialize $(du, dv)^{l,0} = 0$
    Compute $(du, dv)^{l,k}$ by solving Eq. (15)
    Update: $(u, v)_0^{l,k} = (u, v)^{l,k-1} + (du, dv)^{l,k}$
    Weighted median filter (WMF) $(u, v)_0^{l,k}$
    Recompute $(du, dv)^{l,k}$ after WMF, where:
    $(du, dv)_{new}^{l,k} = (u, v)_0^{l,k} - (u, v)^{l,k-1}$
    Update: $(u, v)^{l,k} = (u, v)^{l,k-1} + (du, dv)_{new}^{l,k}$
    Image restoration using our EAC method:
  Solving Eq. (20) to get the restored $({}^{l}I_1, {}^{l}I_2)$
  end
end

Table 2
Quantitative comparison (AAE/AEE/time(s)) of the image restoration methods on sequences from the Middlebury training set.

Method	Nir [30] (IFT)	NNF (NoRest.)	NNF+IFT	NNF+EAC
RubberW.	8.418/0.273/171	2.343/0.073/460	2.459/0.071/740	2.571/0.076/465
Venus	4.161/0.292/159	3.412/0.242/241	3.341/0.238/527	3.177/0.230/245
Dimetr.	3.488/0.186/248	2.653/0.135/456	2.313/0.118/700	2.241/0.115/460
Hydra.	3.127/0.328/242	1.861/0.154/451	1.843/0.151/670	1.879/0.152/455
Grove2	2.616/0.182/285	1.435/0.101/589	1.394/0.097/946	1.321/0.091/595
Grove3	6.728/0.709/282	4.464/0.429/494	4.435/0.431/810	4.348/0.427/498
Urban2	3.454/0.435/277	1.978/0.221/441	1.908/0.208/815	1.850/0.210/448
Urban3	2.343/0.073/278	2.607/0.372/444	2.454/0.352/800	2.287/0.325/450
Norm.Avg.	1.745/1.527/0.54	1.055/1.069/0.99	1.024/1.025/1.66	1.000/1.000/1.00

Table 3
Quantitative comparison (AAE/AEE/time(s)) of the image restoration methods on 3 training sequences from the BlurSequences benchmark [38].

Method	NNF (NoRest.)	NNF+IFT	NNF+EAC
Avg.desert	6.477/0.939/335	7.0080/1.046/603	5.819/0.836/341
Avg.elephant	15.71/4.253/355	15.581/4.262/581	15.52/4.196/363
Avg.market	12.41/2.445/326	12.456/2.469/531	12.36/2.368/333
Norm.Avg.	1.027/1.033/0.977	1.040/1.052/1.657	1.000/1.000/1.00

Table 4
Quantitative comparison (FlowErr/FlowErrOcc/time(s)) of the image restoration methods on the first 20 frames from the training set of KITTI benchmark.

Method	NNF (NoRest.)	NNF+IFT	NNF+EAC
Avg.	0.095/0.198/703	0.099/0.204/1279	0.091/0.195/732
Norm.Avg.	1.022/1.015/0.961	1.088/1.047/1.748	1.000/1.000/1.00

6. Experiments

In this section, we evaluate our experiments quantitatively and visually. Our quantitative evaluations consist of three parts: (1) testing whether our EAC technique is effective and efficient on the training set of benchmarks; (2) testing whether our EAC technique still performs well under heavy noise on the simulated noisy training set of benchmarks; and (3) comparing the performance of our EAC-NNF method with state-of-the-art methods on the testing set of benchmarks. In the visual evaluation part, we also consider three aspects: (1) testing on synthetic sequences; (2) testing on real sequences; and (3) testing the illumination handling ability on KITTI sequences.

6.1. Quantitative evaluation

We use two standard error measures (Average Angle Error (AAE) [50] and Average Endpoint Error (AEE) [51]), two special error measures of KITTI [46] (FlowErr and FlowErrOcc) and the computation time. FlowErr computes the average number of incorrect pixels for all pixels, FlowErrOcc computes the average number of incorrect pixels for all occluded pixels. The angular error (AE) and the endpoint error (EE) are computed via [13]:

$$AE = \arccos\left(\frac{u_T u + v_T v + 1}{\sqrt{(u_T^2 + v_T^2 + 1)(u^2 + v^2 + 1)}}\right) \tag{22}$$

$$EE = \sqrt{(u_T - u)^2 + (v_T - v)^2} \tag{23}$$

where (u, v) is the estimated optical flow and (u_T, v_T) represents the ground truth optical flow.

6.1.1. Results on the training set of benchmarks

We test whether computing optical flow and restoring images simultaneously is effective for improving the flow accuracy. We compare the NNF (NoRestore) method (without image restoration) to the NNF+IFT method (use of the IFT [30] to restore images) and the NNF+EAC method (use of our EAC to restore images).

Table 2 shows the results on the Middlebury training set. With our EAC technique, the AAE/AEE of the NNF (NoRestore) method is improved by about 6% and the computational time is barely increased. In contrast, when applying the IFT technique, the accuracy degrades. When comparing the NNF+IFT method to the NNF (NoRestore) method, the AAE/AEE shows little change, but the computational time increases by about 66.2%. Furthermore, Table 2 shows that our NNF +EAC method improves the accuracy of the original joint method of Nir et al. [30] by about 70% on AAE and 50% on AEE respectively.

Table 3 shows the results on the BlurSequences benchmark. All frames of each sequence (i.e. *desert*, *elephant* and *market*) are selected for evaluation. For example, Avg.desert represents the average result of

Table 5
Quantitative comparison (AAE/AEE/time(s)) of the image restoration methods on nine sequences from the MPI-Sintel training set. From the second column to the fourth column are the results on the clean pass, and from the fifth column to the seventh column are the results on the final pass.

Method	NNF (NoRest.)	NNF+IFT	NNF+EAC	NNF (NoRest.)	NNF+IFT	NNF+EAC
alley2	2.151/0.179/135	2.865/0.217/481	2.063/0.174/140	1.881/0.145/158	2.292/0.166/403	1.887/0.137/162
ambush2	18.37/16.34/173	18.45/16.05/436	18.21/15.84/181	33.96/35.93/208	33.34/35.35/453	33.12/35.14/216
bamboo2	4.530/0.300/123	5.623/0.328/415	4.371/0.295/133	4.867/0.426/126	5.748/0.334/407	4.567/0.304/133
bandage2	5.854/0.672/122	6.939/0.676/381	5.765/0.629/129	12.93/1.967/132	13.58/2.008/400	12.45/1.894/140
cave2	4.845/1.525/108	4.896/1.438/261	4.643/1.381/115	5.129/1.709/109	5.184/1.737/251	5.031/1.519/115
market2	7.586/1.328/100	7.924/1.390/234	7.198/1.304/104	8.892/1.547/91	9.083/1.577/225	8.722/1.483/98
shaman2	2.412/0.138/130	2.492/0.149/265	2.352/0.133/136	2.808/0.157/128	2.719/0.160/286	2.570/0.149/136
sleeping2	1.364/0.061/132	1.482/0.067/277	1.280/0.057/140	1.452/0.062/122	1.787/0.078/269	1.316/0.059/130
temple2	4.554/0.796/100	4.592/0.809/266	4.411/0.765/107	9.250/1.669/94	9.081/1.597/266	9.162/1.588/102
Norm.Avg.	1.027/1.037/0.95	1.099/1.027/2.55	1.000/1.000/1.00	1.030/1.032/0.95	1.050/1.017/2.40	1.000/1.000/1.00

Table 6
Performance (AAE/AEE) of the EAC method on sequences from the Middlebury training set with different baseline methods.

Method	RubberW.	Venus	Dimetr.	Hydra.	Grove2	Grove3	Urban2	Urban3	Norm.Avg.
BA [16]	3.156/0.097	4.752/0.293	4.110/0.199	2.060/0.177	2.492/0.172	6.496/0.660	2.965/0.376	4.728/0.605	1.072/1.070
BA+IFT	4.363/0.131	4.802/0.298	2.293/0.116	2.081/0.177	2.418/0.163	6.505/0.680	2.916/0.367	4.271/0.515	1.033/1.014
BA+EAC	3.399/0.103	4.586/0.286	3.342/0.162	2.169/0.181	2.227/0.152	6.211/0.644	2.657/0.369	4.117/0.516	1.000/1.000
NL [36]	2.354/0.073	3.333/0.237	2.570/0.131	1.828/0.151	1.483/0.103	5.037/0.470	2.088/0.218	2.574/0.379	1.022/1.053
NL+IFT	8.059/0.258	3.433/0.269	3.157/0.153	3.296/0.320	1.969/0.140	5.641/0.554	2.641/0.262	3.014/0.384	1.499/1.395
NL+EAC	2.736/0.083	3.218/0.231	2.213/0.113	1.855/0.151	1.405/0.097	4.970/0.473	1.896/0.200	2.533/0.338	1.000/1.000
NLP [28]	2.351/0.073	3.341/0.238	2.570/0.131	1.829/0.151	1.496/0.104	4.951/0.463	2.048/0.215	2.622/0.393	1.026/1.053
NLP+IFT	7.964/0.255	3.404/0.266	3.183/0.154	3.267/0.322	1.964/0.140	5.461/0.538	2.679/0.264	3.033/0.388	1.497/1.386
NLP+EAC	2.732/0.083	3.164/0.230	2.213/0.113	1.860/0.151	1.416/0.097	4.843/0.465	1.887/0.199	2.568/0.344	1.000/1.000

Table 7
Quantitative comparison (AAE/AEE) of the image restoration methods on sequences from the Middlebury training set with added Gaussian noise. On the top table, from the second column to the fifth column, and from the sixth column to the ninth column, Gaussian noise with deviation $\sigma_n^2 = 5^2$ and $\sigma_n^2 = 10^2$ have been respectively added. On the bottom table, Gaussian noise with deviation $\sigma_n^2 = 30^2$ and $\sigma_n^2 = 50^2$ have been respectively added.

Method	Nir [30] (IFT)	NNF (NoRest.)	NNF+IFT	NNF+EAC	Nir [30] (IFT)	NNF (NoRest.)	NNF+IFT	NNF+EAC
RubberW.	4.763/0.149	3.440/0.103	3.114/0.093	3.144/0.094	4.666/0.140	5.345/0.162	4.585/0.138	4.268/0.127
Venus	3.341/0.238	3.813/0.270	3.732/0.261	3.392/0.250	5.095/0.312	4.447/0.305	4.142/0.310	3.966/0.291
Dimetr.	2.313/0.118	2.482/0.128	2.362/0.122	2.305/0.119	2.519/0.128	3.044/0.154	2.780/0.139	2.647/0.136
Hydra.	1.843/0.151	2.063/0.177	2.020/0.170	2.012/0.168	2.189/0.191	2.525/0.227	2.313/0.206	2.269/0.198
Grove2	2.905/0.207	1.520/0.110	1.508/0.105	1.399/0.096	2.463/0.167	1.731/0.125	1.719/0.121	1.573/0.110
Grove3	7.183/0.731	4.638/0.447	4.682/0.454	4.477/0.443	6.710/0.706	5.085/0.488	5.094/0.492	5.063/0.487
Urban2	3.726/0.449	2.390/0.239	2.242/0.245	2.191/0.231	3.358/0.411	3.130/0.285	2.970/0.279	2.722/0.270
Urban3	5.861/0.652	2.913/0.401	2.597/0.377	2.568/0.356	6.754/0.705	3.568/0.481	3.257/0.439	3.648/0.439
Norm.Avg.	1.486/1.538	1.082/1.069	1.036/1.041	1.000/1.000	1.291/1.342	1.104/1.082	1.027/1.035	1.000/1.000
RubberW.	16.97/0.521	12.51/0.373	11.25/0.319	10.36/0.309	30.989/1.388	18.01/0.584	17.17/0.540	16.26/0.520
Venus	12.72/0.712	7.282/0.526	7.375/0.505	7.207/0.494	20.615/1.028	11.59/0.845	12.82/0.851	9.787/0.716
Dimetr.	10.32/0.552	9.243/0.433	5.891/0.321	6.246/0.331	35.889/1.265	18.57/0.918	19.01/0.889	17.42/0.841
Hydra.	6.969/0.650	4.638/0.442	3.613/0.343	3.685/0.357	14.679/1.413	6.069/0.587	6.042/0.578	6.107/0.596
Grove2	5.666/0.415	3.154/0.232	2.572/0.228	2.705/0.194	16.125/1.226	4.872/0.353	5.111/0.367	6.665/0.373
Grove3	10.13/1.030	7.130/0.687	6.633/0.659	6.801/0.650	18.070/1.601	9.523/0.864	9.836/0.873	9.713/0.889
Urban2	12.51/1.038	7.247/0.623	5.789/0.541	5.409/0.471	16.709/1.016	10.23/0.833	10.09/0.827	10.12/0.812
Urban3	13.31/1.395	7.108/0.857	6.435/0.764	5.506/0.762	19.773/2.330	9.087/1.249	9.392/1.285	9.415/1.282
Norm.Avg.	1.848/1.769	1.217/1.171	1.034/1.032	1.000/1.000	2.022/1.869	1.029/1.035	1.047/1.031	1.000/1.000

the 9 frames of the *desert* sequence. When employing our EAC technique to restore images during the optimization, the total average accuracy improvement is about 3%. For some sequence, such as *desert*, the improvement can even reach 9%. The IFT technique not only performs worse but its running time increases more than 65%.

Table 4 shows the results on the KITTI training sequences (frames 0–19 are selected for testing). Clearly, our EAC technique improves the performance of the NNF (NoRestore) method, while the IFT technique fails to improve the accuracy.

Table 5 shows the results on the MPI-Sintel training sequences. All frames of each sequence are selected for evaluation averagely. Our EAC technique is effective in improving the accuracy on both the clean pass and the final pass. For some sequences, the benefit of EAC is limited while for some other sequences, such as *cave2*, *market2* and *sleeping2*, the increase is approximately 10%.

In addition, we incorporate the IFT technique and the EAC technique into three classical variational algorithms: Black and Anandan (BA) [16], Classic+NL [36] and Classic+NLP [28]. Table 6 shows that the IFT technique is not stable. It slightly improves the performance of the BA algorithm, but it negatively impacts the performance of the Classic+NL [36] algorithm and the Classic+NLP [28] algorithm. The gradient descent optimization, used by the IFT technique, depends on the initialization and the step size and can fall into a local minimum. Our EAC technique is beneficial for all of them. For instance, both AAE and AEE of the BA algorithm are improved by about 7% when our EAC technique is applied.

Summary: From the results of Tables 2–6, it becomes clear that restoring images during optical flow computation is an effective method to improve the flow accuracy. Our EAC technique, which can be seamlessly integrated into variational optical flow models, is effective. In addition, its computational time barely increases due to our proposed minimization approach, and it saves over 60% time compared to the IFT technique of [30].

6.1.2. Results on the training set of benchmarks with added noise

To test whether the EAC technique works well under heavy noise, we add Gaussian noise with $\sigma_n = 5$, $\sigma_n = 10$, $\sigma_n = 30$ and $\sigma_n = 50$ to four datasets. Comparing Tables 7–10 with Tables 2–5, we see that the IFT technique [30] is helpful to improve the flow accuracy. For example, in Tables 7 and 8, the NNF+IFT method performs much better than the NNF (NoRestore) method. More importantly, the improvements from the NNF+EAC method are bigger than the experiments without noise. In Table 2, the improvement of AAE/AEE due to EAC is about 5%, but in Table 7, the improvement of AAE/AEE due to EAC reaches 10.4%/8.2% when $\sigma_n^2 = 10^2$, and increases to 21.7%/17.1% when $\sigma_n^2 = 30^2$. Table 9 shows that our NNF+EAC method performs best in the KITTI benchmark testing set. Especially when comparing our method to the NNF-IFT, the benefits from EAC are significant. Table 10 shows that our NNF+EAC method increases the accuracy with about 4% on the sequences with Gaussian noise with $\sigma_n^2 = 5^2$, $\sigma_n^2 = 10^2$, $\sigma_n^2 = 30^2$ and

Table 8
Quantitative comparison (AAE/AEE) of the restoration methods on 3 training sequences from the BlurSequences benchmark with added Gaussian noise. On the top table, from the second column to the fourth column, and from the fifth column to the seventh column, Gaussian noise with deviation $\sigma_n^2 = 5^2$ and $\sigma_n^2 = 10^2$ have been respectively added. On the bottom table, Gaussian noise with deviation $\sigma_n^2 = 30^2$ and $\sigma_n^2 = 50^2$ have been respectively added.

Method	NNF(NoRest.)	NNF+IFT	NNF+EAC	NNF(NoRest.)	NNF+IFT	NNF+EAC
Avg.desert	7.7910/1.170	7.581/1.084	7.157/1.063	9.574/1.306	9.836/1.240	9.193/1.148
Avg.elephant	15.71/4.290	15.42/4.221	15.41/4.181	15.94/4.154	15.39/4.173	15.38/4.163
Avg.market	12.58/2.433	12.44/2.460	12.44/2.420	13.35/2.690	13.00/2.682	12.87/2.526
Norm.Avg.	1.031/1.032	1.013/1.014	1.000/1.000	1.038/1.040	1.021/1.033	1.000/1.000
Avg.desert	19.76/2.559	16.85/2.296	13.98/1.805	33.03/3.882	32.57/3.706	29.82/3.551
Avg.elephant	15.92/4.309	15.84/4.385	15.54/4.296	16.21/4.558	16.09/4.495	15.90/4.281
Avg.market	12.97/2.522	12.80/2.569	12.86/2.493	13.77/2.711	13.30/2.697	12.86/2.595
Norm.Avg.	1.148/1.093	1.073/1.076	1.000/1.000	1.076/1.069	1.058/1.045	1.000/1.000

Table 9
Quantitative comparison (FlowErr/FlowErrOcc) of the restoration methods on the first 20 frames from the KITTI training sequences with added Gaussian noise. On the top table, from the second column to the fourth column, and from the fifth column to the seventh column, Gaussian noise with deviation $\sigma_n^2 = 5^2$ and $\sigma_n^2 = 10^2$ have been respectively added. On the bottom table, Gaussian noise with deviation $\sigma_n^2 = 30^2$ and $\sigma_n^2 = 50^2$ have been respectively added.

Method	NNF (NoRest.)	NNF +IFT	NNF +EAC	NNF (NoRest.)	NNF +IFT	NNF +EAC
Avg.	0.100/ 0.206	0.098/ 0.205	0.096/ 0.197	0.109/ 0.212	0.110/ 0.214	0.107/ 0.208
Norm.Avg.	1.042/ 1.046	1.021/ 1.041	1.000/ 1.000	1.019/ 1.015	1.028/ 1.029	1.000/ 1.000
Avg.	0.152/ 0.251	0.150/ 0.250	0.146/ 0.245	0.223/ 0.321	0.219/ 0.318	0.207/ 0.300
Norm.Avg.	1.041/ 1.024	1.027/ 1.020	1.000/ 1.000	1.077/ 1.070	1.058/ 1.060	1.000/ 1.000

$\sigma_n^2 = 50^2$. On the other side, the performance of the IFT technique is again not stable. It performs poor for most sequences, but for some sequences, e.g., Gaussian noise with deviation $\sigma_n^2 = 10^2$ and $\sigma_n^2 = 30^2$, it performs well. This is because the gradient descent method used in IFT heavily depends on the initialization and the step size. For different sequences, the appropriate initialization and step size are hard to select.

In Table 11, we quantitatively compare the signal-noise-ratio (SNR) of the input image (frame 10) before and after the EAC image restoration. Table 11 shows that our EAC method is effectively reduces noise in sequences with or without heavy noise, but the noise reduction is not very large (about 3%). In addition, Fig. 1 shows the visual results of sequences *Grove2* and *Urban2*. The edges are well preserved when using our EAC technique for denoising restoration.

Summary: From the results of Tables 7–11, we observe that under heavy noise, restoring images during optical flow computation plays a significant role in improving the flow accuracy. Our EAC technique produces more accurate optical flow than the IFT method [30].

6.1.3. Results of method comparisons and benchmark evaluations

We quantitatively evaluate our methods in two aspects: (1) comparing our algorithm with the related methods and (2) evaluating our algorithm on the testing set of the famous benchmarks.

(1) *Method comparisons*: We compare our NNF-EAC method to five state-of-the-art variational methods: Corr.Flow [35], MDP-Flow [1], Classic+NL [36], Classic+NLP [28] and NN-field [21]. Especially the last three methods have a similar objective energy function and optimization process. The NN-field [21] improves the accuracy of methods Classic+NL [36] and Classic+NLP [28] due to the contribution of the approximate NNF, which provides useful correspondence information. Our NNF+EAC method further modifies the NN-field [21] method because it is able to compute optical flow and restore images with preserved edges simultaneously.

On the Middlebury training set (see Table 12), the Corr.Flow [35] and MDP-Flow [1] methods perform worse. Moreover, the three closely related methods, i.e., the Classic+NL [36], Classic+NLP [28] and NN-field [21] methods, achieve equal result, and our NNF+EAC method outperforms them by about 7%.

On the BlurSequences benchmark (see Table 13), the accuracy of the Corr.Flow [35] is much lower. The MDP-Flow [1] achieves better results than Corr.Flow [35]. However, when comparing it with the other four methods, its performance is inferior. Our NNF+EAC method outperforms the other three methods by about 3%. This implies that it can obtain desirable results under motion blur.

On the KITTI benchmark training set (Table 14), our NNF+EAC method performs well. Comparing it to Corr.Flow [35] and MDP-Flow [1], the improvements coming from the NNF+EAC method are high. Compared to the Classic+NL [36] and NN-field [21], the FlowErr and the FlowErrOcc of our method are reduced by over 60% and 20%, respectively.

On the MIT dataset (see Table 15), we choose the heavy noise sequences *fish* (frames 136–150) and *cameramotion* (frames 16–30) for evaluation. The Corr.Flow [35] and MDP-Flow [1] methods still perform much worse than the other four methods. In particular, the AEE of Corr.Flow is about 50% worse than our NNF-EAC method, and the AAE/AEE of MDP-Flow [1] are 22.6%/17.9% worse than our NNF-EAC method.

Summary: From results of Tables 12–15, we observe that our NNF-EAC method performs better than the related state-of-the-art methods in most cases due to the proposed EAC technique.

(2) *Benchmark evaluations*: We evaluate our NNF-EAC method on the testing set of the Middlebury benchmark (results available at http://vision.middlebury.edu/flow/eval/), the KITTI benchmark (results available at http://www.cvlibs.net/datasets/kitti/eval_stereo_flow.php?benchmark=flow) and MPI-Sintel benchmark (results available at http://sintel.is.tue.mpg.de/results).

As shown in Fig. 2, we achieve the topmost performance on the Middlebury testing set, ranking 9th among the tested 121 methods. In particular, for the two heavy noise real sequences *Backyard* and *Basketball*, our method performs best. For example, in the Average Interpolation Error rank list, our method ranks first at disc on *Basketball*; in the Average Normalized Interpolation Error rank list, our method ranks first at untext on *Backyard*, and ranks first at all and untext items on *Basketball*. Comparing our NNF-EAC method to Corr.Flow [35] and MLDP_OF [20], the latter two are good at dealing with illumination changes, but our method surpasses them significantly. More details of our results can be found on the evaluation website of the Middlebury benchmark.

The KITTI benchmark testing set contains 195 pairs of real-world

Table 10
Quantitative comparison (AAE/AEE) of the image restoration methods on sequences from the MPI-Sintel training set with added Gaussian noise. Where Gaussian noise with deviation $\sigma_n^2 = 5^2$, $\sigma_n^2 = 10^2$, $\sigma_n^2 = 30^2$ and $\sigma_n^2 = 50^2$ have been respectively added to tables from top to bottom.

Method	NNF (NoRest.)	NNF+IFT	NNF+EAC	NNF (NoRest.)	NNF+IFT	NNF+EAC
alley2	2.921/0.206	2.658/0.203	2.611/0.198	2.745/0.172	2.441/0.159	2.503/0.165
ambush2	18.34/16.06	18.55/16.10	18.10/15.57	33.89/35.85	34.15/35.89	32.82/35.04
bamboo2	5.164/0.315	5.431/0.323	5.014/0.303	5.646/0.330	5.624/0.334	5.493/0.306
bandage2	6.644/0.661	6.772/0.679	6.605/0.631	15.08/1.977	14.14/1.959	14.26/1.885
cave2	5.028/1.463	5.152/1.487	5.017/1.470	5.135/1.667	5.142/1.663	5.045/1.664
market2	8.551/1.347	8.147/1.343	8.167/1.352	10.85/1.607	9.848/1.571	10.33/1.589
shaman2	2.740/0.127	2.515/0.151	2.582/0.152	3.351/0.175	2.732/0.161	3.030/0.167
sleeping2	1.686/0.073	1.469/0.066	1.578/0.067	2.047/0.088	1.718/0.076	1.886/0.081
temple2	4.745/0.857	4.710/0.846	4.677/0.833	10.31/1.764	10.34/1.771	10.28/1.748
Norm.Avg.	1.027/1.026	1.020/1.030	1.000/1.000	1.040/1.023	1.006/1.022	1.000/1.000
alley2	3.951/0.242	3.070/0.217	23.56/0.236	3.801/0.210	3.042/0.184	3.447/0.202
ambush2	18.51/16.25	18.84/16.33	17.92/16.07	34.29/36.15	34.11/36.12	33.99/35.04
bamboo2	5.892/0.333	5.586/0.329	5.719/0.311	6.712/0.354	5.932/0.338	6.554/0.310
bandage2	7.676/0.707	7.430/0.704	7.395/0.682	17.07/2.010	16.46/2.006	16.50/1.958
cave2	5.337/1.552	5.258/1.535	5.244/1.547	5.422/1.748	5.390/1.728	5.367/1.726
market2	10.94/1.401	9.944/1.373	10.42/1.406	14.10/1.672	13.09/1.649	13.57/1.672
shaman2	3.584/0.195	3.229/0.184	3.300/0.184	4.681/0.231	4.142/0.213	4.409/0.223
sleeping2	2.215/0.094	1.849/0.080	2.131/0.087	2.901/0.123	2.415/0.105	2.728/0.116
temple2	5.117/0.921	5.137/0.932	5.167/0.950	11.56/1.904	11.17/1.882	11.40/1.889
Norm.Avg.	1.039/1.043	0.992/1.010	1.000/1.000	1.026/1.029	0.978/1.025	1.000/1.000
alley2	7.288/0.367	6.542/0.338	6.933/0.356	7.926/0.375	7.239/0.347	7.375/0.364
ambush2	19.64/17.02	19.50/16.98	19.09/16.39	34.79/36.40	34.35/36.41	34.34/36.03
bamboo2	9.627/0.411	8.311/0.384	9.007/0.398	12.24/0.476	10.57/0.440	11.24/0.458
bandage2	11.79/0.876	11.34/0.871	11.39/0.808	21.73/2.181	21.50/2.147	20.96/2.056
cave2	6.282/1.886	6.306/1.897	6.309/1.898	6.526/2.182	6.357/2.138	6.345/2.160
market2	19.89/1.715	18.47/1.670	18.58/1.688	24.83/2.042	23.35/1.991	23.82/2.013
shaman2	8.214/0.416	8.098/0.406	7.855/0.409	10.85/0.535	10.56/0.509	10.30/0.506
sleeping2	4.767/0.199	4.339/0.177	4.307/0.179	7.094/0.301	6.446/0.278	6.548/0.285
temple2	7.049/1.348	6.875/1.288	6.882/1.302	13.95/2.326	14.01/2.309	14.47/2.364
Norm.Avg.	1.046/1.034	0.994/1.025	1.000/1.000	1.033/1.035	0.993/1.029	1.000/1.000
alley2	10.28/0.491	9.627/0.461	9.775/0.478	11.46/0.527	10.94/0.501	11.05/0.450
ambush2	20.33/17.49	19.66/17.32	19.76/16.85	35.01/36.56	35.01/36.40	34.76/35.04
bamboo2	13.80/0.497	12.42/0.470	12.51/0.473	17.21/0.592	15.74/0.563	16.50/0.566
bandage2	15.82/1.065	15.18/1.034	15.35/0.981	24.25/2.275	24.00/2.277	24.33/2.150
cave2	7.356/2.292	7.243/2.251	7.118/2.256	7.458/2.620	7.414/2.594	7.262/2.582
market2	25.57/1.993	24.41/1.921	25.25/1.984	30.48/2.292	30.01/2.259	28.95/2.249
shaman2	12.78/0.643	11.69/0.612	11.96/0.624	15.90/0.818	16.15/0.819	15.22/0.794
sleeping2	7.654/0.322	6.917/0.296	7.159/0.310	11.08/0.490	10.77/0.467	10.60/0.468
temple2	8.697/1.627	8.984/1.682	8.833/1.629	15.61/2.538	16.05/2.561	15.24/2.492
Norm.Avg.	1.039/1.059	0.987/1.018	1.000/1.000	1.028/1.041	1.014/1.035	1.000/1.000

Table 11
SNR of the input images on the Middlebury training set before and after EAC.

SNR	RubberW.	Venus	Dimetr.	Hydra.	Grove2	Grove3	Urban2	Urban3
SNR (before restore)	8.189	7.457	8.750	8.140	6.803	6.732	8.152	7.405
SNR (after restore)	8.435	7.621	8.903	8.308	7.088	6.929	8.298	7.657
$\sigma_n^2 = 10^2$	RubberW.	Venus	Dimetr.	Hydra.	Grove2	Grove3	Urban2	Urban3
SNR (before restore)	8.118	7.451	8.599	8.071	6.764	6.673	7.798	7.325
SNR (after restore)	8.459	7.653	8.898	8.318	7.035	6.892	8.032	7.676
$\sigma_n^2 = 30^2$	RubberW.	Venus	Dimetr.	Hydra.	Grove2	Grove3	Urban2	Urban3
SNR (before restore)	7.579	6.928	7.580	7.484	6.497	6.343	7.073	6.892
SNR (after restore)	7.852	7.199	7.948	7.759	6.720	6.567	7.323	7.185

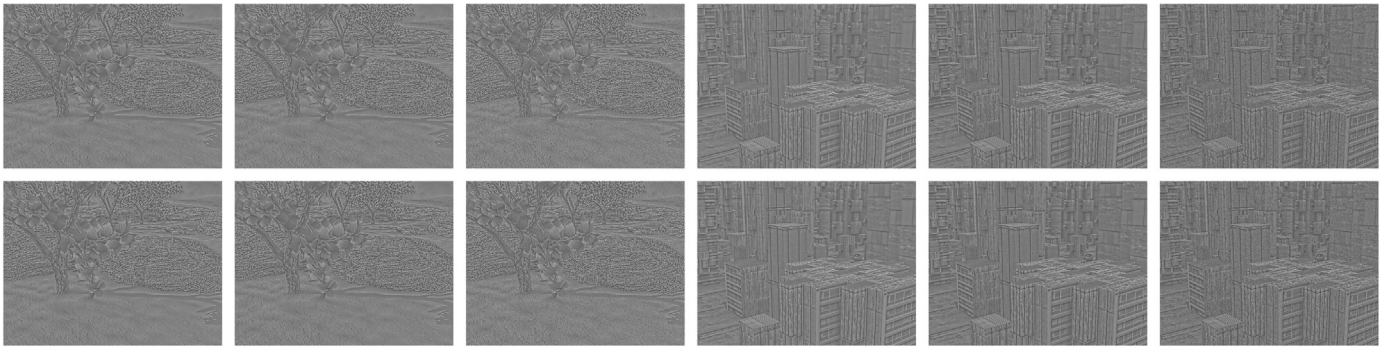

Fig. 1. Visual comparisons on the Middlebury benchmark training set (with structure–texture decomposition). *From top to bottom:* input image (frame 10) before the EAC restoration, and input image after the EAC restoration. *From left to right:* results of *Grove2* without added Gaussian noise, with added Gaussian noise with deviation $\sigma_n^2 = 10^2$ and $\sigma_n^2 = 30^2$, results of *Urban2* without added Gaussian noise, with added Gaussian noise with deviation $\sigma_n^2 = 10^2$ and $\sigma_n^2 = 30^2$.

sequences taken from a driving platform. It includes non-Lambertian surfaces, different lighting conditions, large displacements and different levels of noise. Besides, there are multiple objects in the scene. Thus, motion boundaries can be easily smoothed by flow field denoising or image field filtering. As shown in Table 16, our method outperforms both baseline methods Classic+NL [36] and NNF-Local [44]. This demonstrates that the proposed EAC is effective in denoising the real sequences to improve the accuracy of flow estimation. Our NNF-EAC is competitive to the MLDP_OF [20]. The Out-Noc and Out-All for our method are worse than MLDP_OF [20], but the Avg-Noc and Avg-All are better. Out-Noc denotes the percentage of erroneous pixels in non-occluded areas, Out-All represents the percentage of erroneous pixels in total, Avg-Noc denotes the average disparity divided by the end-point error in non-occluded areas, Avg-All represents the average disparity divided by the end-point error in total [46].

On the MPI-Sintel benchmark testing set, our method outperforms related dense variational optical flow methods (see Table 17), except NNF-Local [44]. NNF-Local [44] uses a different matching approach than their NN-field method [21], while we use the same matching methods as NN-field.

Some methods only perform well on one dataset. For example, the NLTGV-SC [19] performs good on the KITTI benchmark testing set while poorly on the MPI-Sintel benchmark testing set, and the NNF-Local [44] performs much better on the Middlebury benchmark testing set than on the KITTI benchmark testing set. Our method performs equally well on all three benchmarks testing set.

6.2. Visual evaluation

Visually evaluation is carried out in three aspects to test the performance of the proposed EAC-NNF technique in this section.

6.2.1. Evaluation on synthetic sequences

First, we use four representative synthetic sequences with ground-truths are selected from the Middlebury benchmark: *Dimetrodon* (Hidden Texture), *Venus* (Stereo), *Grove3* and *Urban3* (Synthetic). Figs. 3 and 4 show that applying EAC to the NNF method during optimization is not only beneficial to the accuracy but also the edges are better preserved. For *Dimetrodon*, the mouth of the crocodile in the NNF (NoRestore) is too dim to discern. The mouth of the crocodile in the NNF+IFT is clear, but its backgrounds are not correct. In contrast, both the mouth of the crocodile and its background are well restored with NNF+EAC. The two red rectangles on the right show that our estimated flow approximates the ground-truth. For *Venus*, the benefits from our EAC technique are apparent. Many edge flow vectors in NNF (NoRestore) and NNF+IFT are incorrect. Some inaccurate edge flow vectors are corrected in NNF+EAC.

For *Grove3*, our EAC technique is helpful in preserving the edges of small-scale objects, e.g. the twigs and leaves. In particular, for the second red rectangle, the twigs of NNF (NoRestore) and NNF+IFT are not recovered, while it is partially estimated with NNF+EAC. For the fourth red rectangle, some flow boundaries are flattened without image restoration and IFT blurs flow boundaries. Our EAC technique improves their performance. For *Urban3*, NNF+EAC results in fewer incorrect flow vectors around edges than NNF (NoRestore) and NNF +IFT. For example, flow boundaries in the biggest red rectangle of the NNF+EAC are preserved much better. The method of Nir et al. [30] is unable to preserve the motion boundaries and also poorly restores images. The captured boundaries, except in the red rectangle regions, but also in the rest area, are too dim (see the second column of Fig. 4)). [45] implements deblurring to reduce motion blur to the images but useful information in the images is also removed. Recovered motion boundaries are distorted, as can be seen in the first column of Fig. 4).

Second, we test on the synthetic *fish* sequence from the MIT benchmark. It contains a significant amount of noise distributed in the background around the fishes. Fig. 5 shows that our EAC integrated NNF+EAC is superior to the methods of Portz et al. [45], Nir et al. [30], NNF (NoRestore) and NNF+IFT. Similar to the results in Fig. 4, the

Table 12
Quantitative comparison (AAE/AEE) of our NNF-EAC method with five related variational methods on sequences from the Middlebury training set.

Method	Corr.Flow [35]	MDP-Flow [1]	Classic+NL [36]	Classic+NLP [28]	NN-field [21]	NNF-EAC
RubberW.	3.510/0.235	2.519/0.082	2.354/0.073	2.351/0.073	2.395/0.074	2.553/0.078
Venus	4.517/0.373	3.246/0.221	3.333/0.237	3.341/0.238	3.177/0.231	3.217/0.234
Dimetrodon	6.254/0.318	3.023/0.152	2.570/0.131	2.570/0.131	2.929/0.148	2.241/0.115
Hydrangea	2.224/0.196	1.990/0.164	1.829/0.151	1.828/0.151	1.858/0.152	1.837/0.151
Grove2	2.610/0.231	1.798/0.128	1.483/0.103	1.496/0.104	1.420/0.102	1.331/0.092
Grove3	6.505/0.687	4.879/0.463	5.037/0.470	4.951/0.463	4.376/0.425	4.351/0.429
Urban2	3.887/0.399	1.885/0.181	2.088/0.218	2.048/0.215	2.079/0.253	1.894/0.231
Urban3	6.220/0.853	4.708/0.469	2.688/0.376	2.622/0.393	3.134/0.557	2.411/0.328
Norm.Avg.	1.802/1.990	1.213/1.126	1.078/1.063	1.070/1.068	1.078/1.174	1.000/1.000

Table 13
Quantitative comparison (AAE/AEE) of our NNF-EAC method with five related methods on 3 training sequences from the BlurSequences benchmark.

Method	Corr.Flow [35]	MDP-Flow [1]	Classic+NL [36]	Classic+NLP [28]	NN-field [21]	NNF-EAC
Avg.desert	10.59/1.586	9.303/1.361	6.491/0.903	6.422/0.901	6.704/1.032	5.819/0.836
Avg.elephant	17.12/4.884	15.66/4.193	15.62/4.241	15.61/4.237	15.41/4.171	15.52/4.196
Avg.market	16.83/4.004	12.70/2.245	12.59/2.526	12.46/2.476	12.23/2.336	12.37/2.368
Norm.Avg.	1.322/1.416	1.117/1.054	1.030/1.037	1.024/1.030	1.021/1.020	1.000/1.000

Table 14
Quantitative comparison (FlowErr/FlowErrOcc) of our NNF-EAC method with five related methods on the first 20 frames from the training set of KITTI benchmark.

Method	Corr. Flow [35]	MDP-Flow [1]	Classic +NL [36]	Classic +NLP [28]	NN-field [21]	NNF-EAC
Avg.	0.319/0.356	0.192/0.286	0.146/0.244	0.095/0.199	0.151/0.248	0.091/0.195
Norm. Avg.	3.504/1.826	2.110/1.467	1.604/1.252	1.044/1.021	1.659/1.272	1.000/1.000

Table 15
Quantitative comparison (AAE/AEE) of our NNF-EAC method with five related variational methods on 2 training sequences from the MIT benchmark [47].

Method	Corr. Flow [35]	MDP-Flow [1]	Classic +NL [36]	Classic +NLP [28]	NN-field [21]	NNF-EAC
Avg.fish	15.73/0.848	22.167/0.801	21.397/0.795	21.327/0.790	18.609/0.675	18.567/0.673
Avg.camer.	8.864/0.751	7.889/0.465	6.826/0.442	6.836/0.442	6.870/0.436	5.952/0.401
Norm.Avg.	1.003/1.489	1.226/1.179	1.151/1.153	1.149/1.147	1.040/1.036	1.000/1.000

Table 16
Ranking of optical flow methods on the KITTI flow2012 test set (error threshold 3 pixels).

Method	Out-Noc (%)	Out-All (%)	Avg-Noc (px)	Avg-All (px)
NLTGV-SC [19]	5.93	11.96	1.6	3.8
MLDP-OF [20]	8.67	18.78	2.4	6.7
NNF-EAC	9.72	19.56	1.7	4.7
ROF-NND [43]	10.44	21.23	2.5	6.5
Classic+NL [36]	10.49	20.64	2.8	7.2
NNF-Local [44]	10.68	21.09	2.7	7.4

Table 17
AEE ranking of optical flow methods on the MPI-Sintel test set.

Method	Clean	Final
ROF-NND [43]	8.061	9.286
Classic+NL [36]	7.961	9.153
NLTGV-SC [19]	7.680	8.746
Classic+NLP [28]	5.837	8.445
MDP-Flow2 [1]	6.731	8.291
MLDP-OF [20]	7.297	8.287
WLIF-Flow [25]	5.734	8.049
PatchWMF-OF [29]	5.550	7.971
NNF-EAC[1]	**5.468**	**7.674**
NNF-Local [44]	5.386	7.249

[1] The bold values is our method.

estimated edges of [45] are blurry. The outlines with Nir et al. [30] are too dim to be discerned. The noise in the estimated flow field of NNF (NoRestore) is higher than both NNF+IFT and NNF+EAC. Edges are badly recovered. In contrast, noise is significantly reduced with NNF +IFT and the NNF+EAC. However, due to the inability of IFT to preserve edges, some flow boundaries are incorrectly restored. EAC is not only good at denoising but also good at preserving edges. The lost fins (second red rectangle) and the dim tail (third red rectangle) of NNF +IFT are distinguished with NNF+EAC.

Third, we test on the synthetic *desert* sequence from BlurSequences. This sequence contains complex spatially-varying motion blur. As shown in Fig. 6, the recovered flow boundaries of the eagle of the methods of Nir et al. [30], NNF (NoRestore) and NNF+IFT are far from the ground-truth. In NNF (NoRestore) and NNF+IFT, the shape of the estimated right wing is significantly different from the ground truth. In our NNF+EAC, this is partially solved. Although some information of the right wing and the head is still not recovered accurately, at least their shapes are approximate to the real objects.

Summary: Both the IFT technique of [30] and our EAC technique can restore images during minimization, but the IFT technique cannot preserve edges as well as EAC.

6.2.2. *Evaluation on real sequences*

Three real sequences (*Backyard*, *Basketball* and *Dumptruck*, see Fig. 7) without ground-truth are selected from the Middlebury benchmark. For *Backyard*, the right foot of the boy is incorrectly estimated in both [45] and [30], disappears in NNF (NoRestore), is partially recovered in NNF+IFT, and almost completely restored in our NNF +EAC. The hair of the girl on the right is inaccurately recovered in [45,30], NNF (NoRestore) and NNF+IFT. EAC improves the accuracy. For *Basketball*, the recovered shapes of the basketball and the head of the man on the right are close to the real objects. However, these shapes in [45], Nir et al. [30], NNF (NoRestore) and NNF+IFT are difficult to distinguish. Especially for Nir et al. [30], the edges nearly disappear. For *Dumptruck*, edges caused by the pole are sharp in the

Average endpoint error	avg. rank	Army (Hidden texture) GT im0 im1 all	disc	untext	Mequon (Hidden texture) GT im0 im1 all	disc	untext	Schefflera (Hidden texture) GT im0 im1 all	disc	untext	Wooden (Hidden texture) GT im0 im1 all	disc	untext	Grove (Synthetic) GT im0 im1 all	disc	untext	Urban (Synthetic) GT im0 im1 all	disc	untext	Yosemite (Synthetic) GT im0 im1 all	disc	untext	Teddy (Stereo) GT im0 im1 all	disc	untext
MDP-Flow2 [68]	9.8	0.08 8	0.21 4	0.07 15	0.15 2	**0.48** 1	**0.11** 1	0.20 5	0.40 5	0.14 2	0.15 20	0.80 35	0.08 11	0.63 19	0.93 19	0.43 20	0.26 4	0.76 4	0.23 7	0.11 14	0.12 10	0.17 13	0.38 6	0.79 5	0.44 6
NN-field [71]	10.9	0.08 8	0.22 16	**0.05** 1	0.17 8	0.55 10	0.13 11	0.19 3	0.39 4	0.15 7	**0.09** 1	0.48 4	**0.05** 1	**0.41** 1	**0.61** 1	**0.20** 1	0.52 56	**0.64** 1	0.26 15	0.13 38	0.13 32	0.20 30	0.35 3	0.83 8	**0.21** 1
WLIF-Flow [93]	18.5	0.08 8	0.21 4	0.06 6	0.18 11	0.55 10	0.15 22	0.25 18	0.56 20	0.17 15	0.14 8	0.68 9	0.08 11	0.61 16	0.91 17	0.41 18	0.43 30	0.96 15	0.29 24	0.13 38	0.12 10	0.21 36	0.51 32	1.03 33	0.72 34
NNF-EAC [104]	19.8	0.09 30	0.22 16	0.07 15	0.17 8	0.53 6	0.13 11	0.23 11	0.49 12	0.15 7	0.16 34	0.80 35	0.09 25	0.60 13	0.89 13	0.40 16	0.38 20	0.78 6	0.28 18	0.12 27	0.12 10	0.18 22	0.57 43	1.24 47	0.69 29
Correlation Flow [75]	24.8	0.09 30	0.23 23	0.07 15	0.17 8	0.58 15	**0.11** 1	0.43 60	0.99 62	0.15 7	0.11 4	0.47 3	0.08 11	0.75 40	1.08 40	0.56 42	0.41 26	0.92 13	0.30 28	0.14 48	0.13 32	0.27 65	0.40 8	0.85 9	0.42 5
Classic+NL [31]	33.5	0.08 8	0.23 23	0.07 15	0.22 37	0.74 49	0.18 44	0.29 30	0.65 28	0.19 37	0.15 20	0.73 21	0.09 25	0.64 22	0.93 19	0.47 25	0.52 56	1.12 34	0.33 43	0.16 73	0.13 32	0.29 75	0.49 23	0.98 23	0.74 43
MLDP_OF [89]	40.3	0.11 54	0.28 51	0.09 60	0.18 11	0.56 13	0.13 11	0.34 45	0.79 46	0.17 15	0.16 34	0.82 38	0.09 25	0.72 36	1.05 37	0.50 32	0.34 15	1.10 32	0.27 17	0.18 89	0.15 65	0.44 107	0.76 67	1.09 39	0.69 29
ROF-ND [109]	52.2	0.12 74	0.29 58	0.09 60	0.26 68	0.72 42	0.17 33	0.36 49	0.86 50	0.17 15	0.14 8	0.46 2	0.12 58	0.83 52	1.18 50	0.69 55	0.50 51	1.15 38	0.35 53	0.21 102	0.17 83	0.36 97	0.69 59	1.40 54	0.74 43

Fig. 2. AEE ranking of optical flow methods on the Middlebury test set.

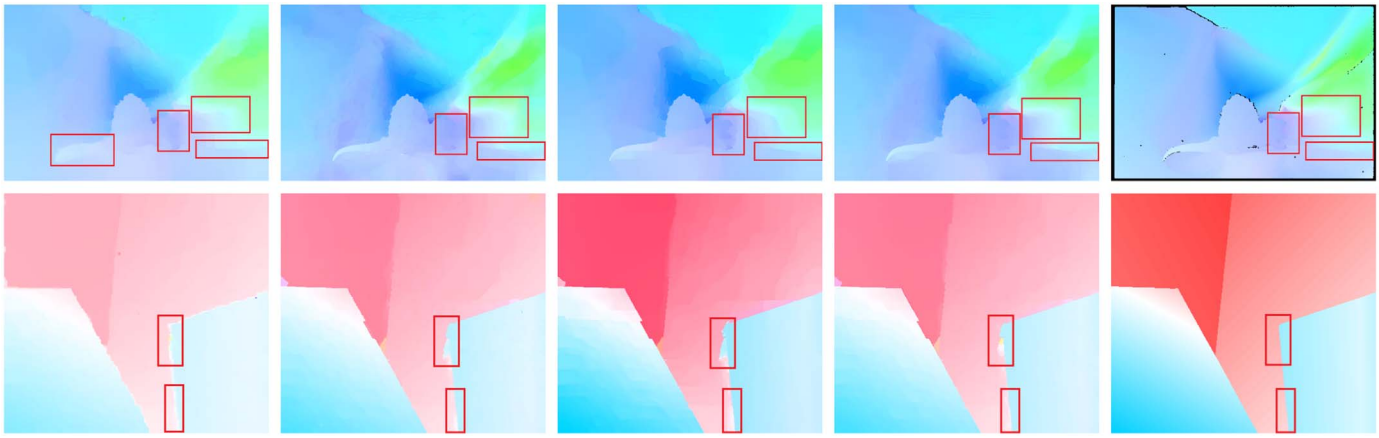

Fig. 3. Visual comparisons on the Middlebury benchmark training set. *From top to bottom:* sequences *Dimetrodon* (Hidden Texture) and *Venus* (Stereo). *From left to right:* results of Nir et al. [30], NNF (NoRestore), NNF+IFT and NNF+EAC, and the corresponding ground-truths. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

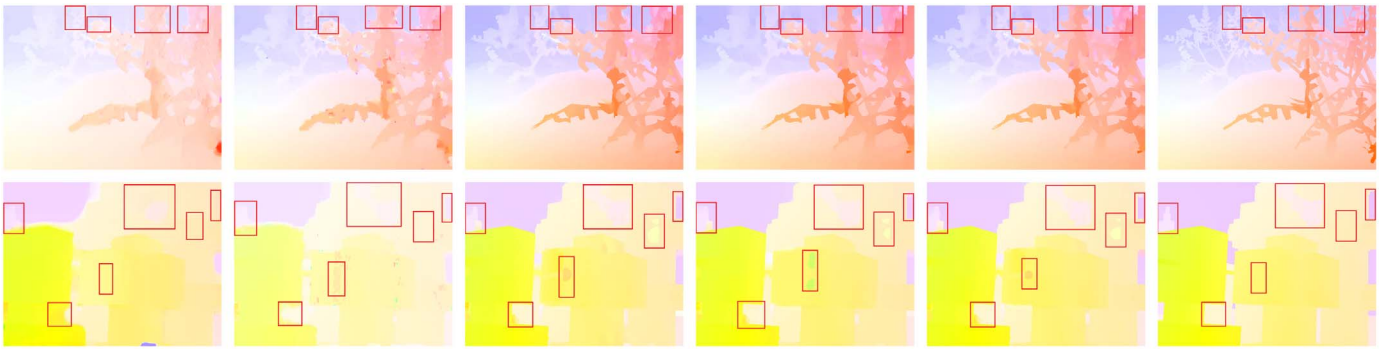

Fig. 4. Visual comparisons on the Middlebury benchmark training set. *From top to bottom:* sequences *Grove3* and *Urban3* (Synthetic). *From left to right:* results of Portz et al. [45], Nir et al. [30], NNF (NoRestore), NNF+IFT and NNF+EAC, and the corresponding ground-truths. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

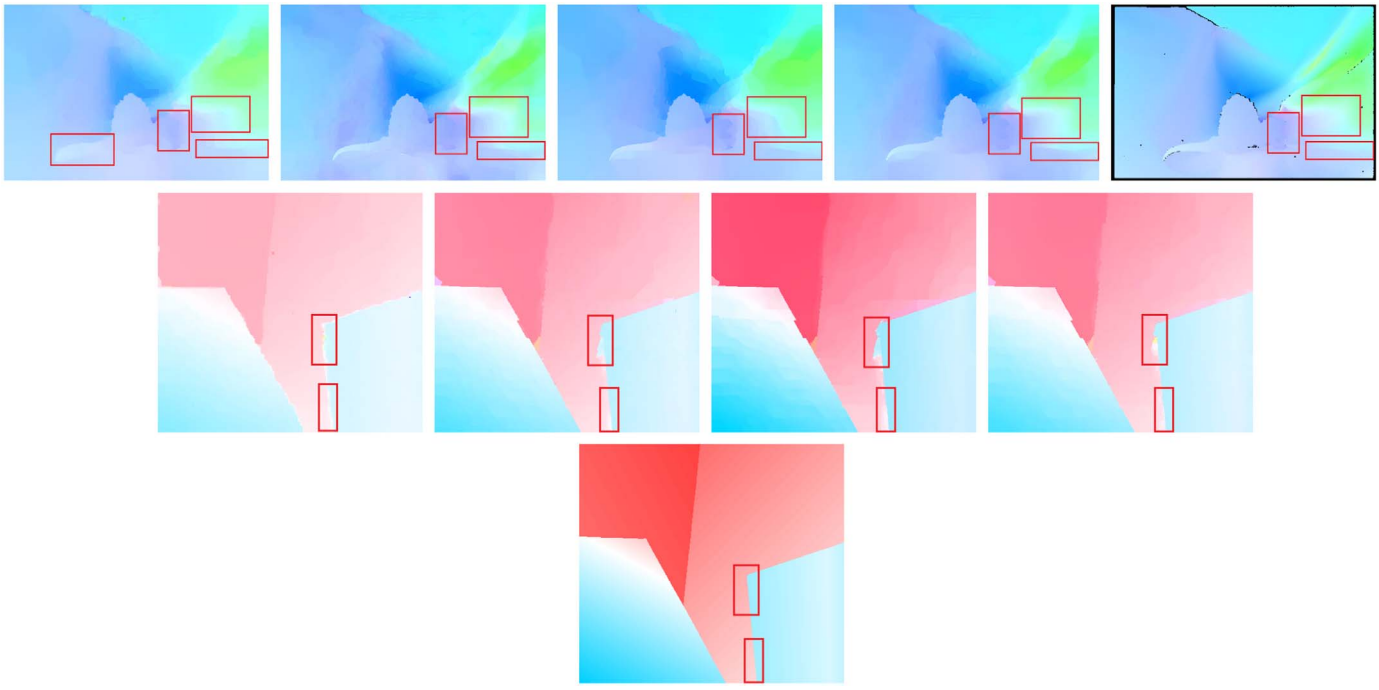

Fig. 5. Visual comparisons of the *fish* sequence on the MIT benchmark. *From left to right:* results of Portz et al. [45], Nir et al. [30], NNF (NoRestore), NNF+IFT and NNF+EAC, and the corresponding ground-truths.

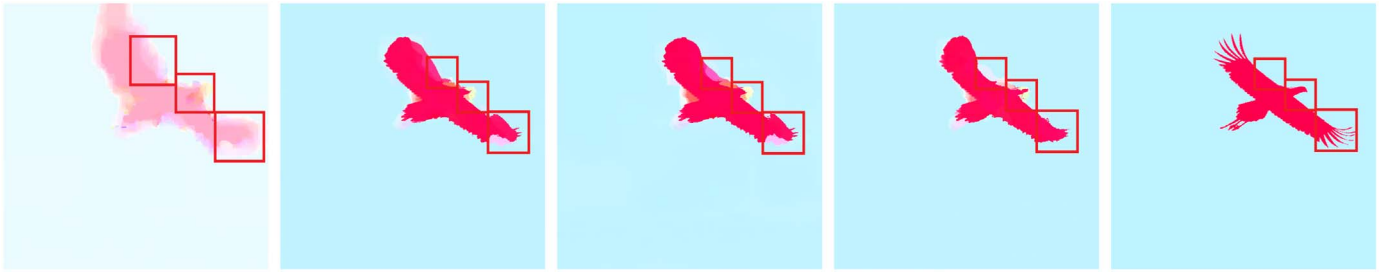

Fig. 6. Visual comparisons of the *desert* sequence on the BlurSequences benchmark. *From left to right:* results of Nir et al. [30], NNF (NoRestore), NNF+IFT and NNF+EAC, and the corresponding ground-truths.

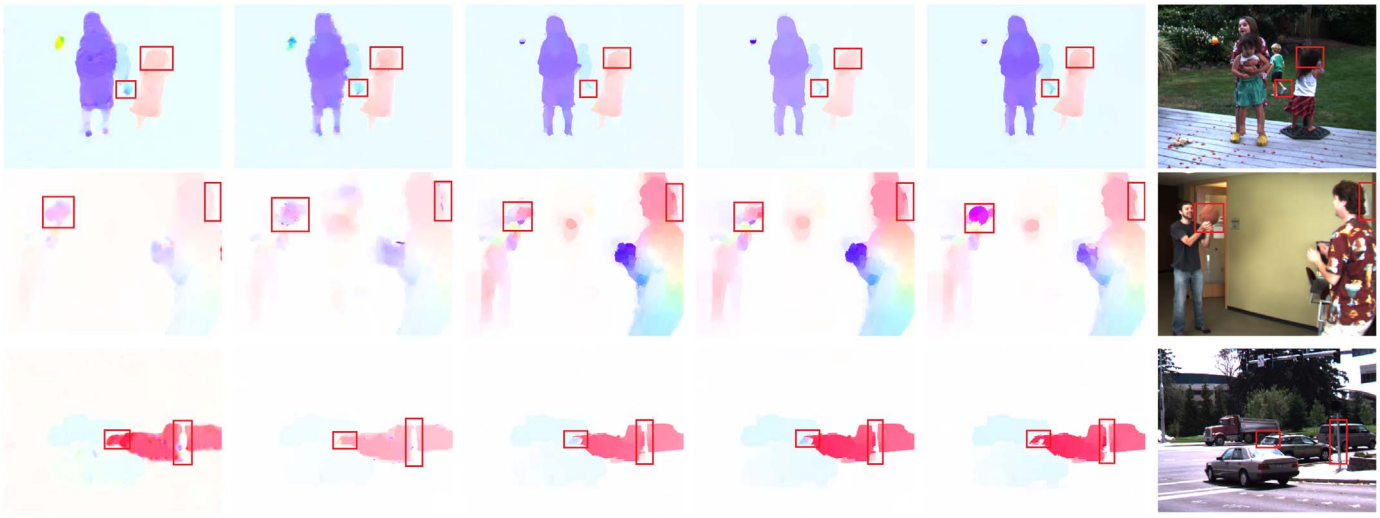

Fig. 7. Visual comparisons on the Middlebury benchmark training set. *From top to bottom:* real sequences *Backyard*, *Basketball* and *Dumptruck*. *From left to right:* results of Portz et al. [45], Nir et al. [30], NNF (NoRestore), NNF+IFT and NNF+EAC, and the corresponding input images (frame 10).

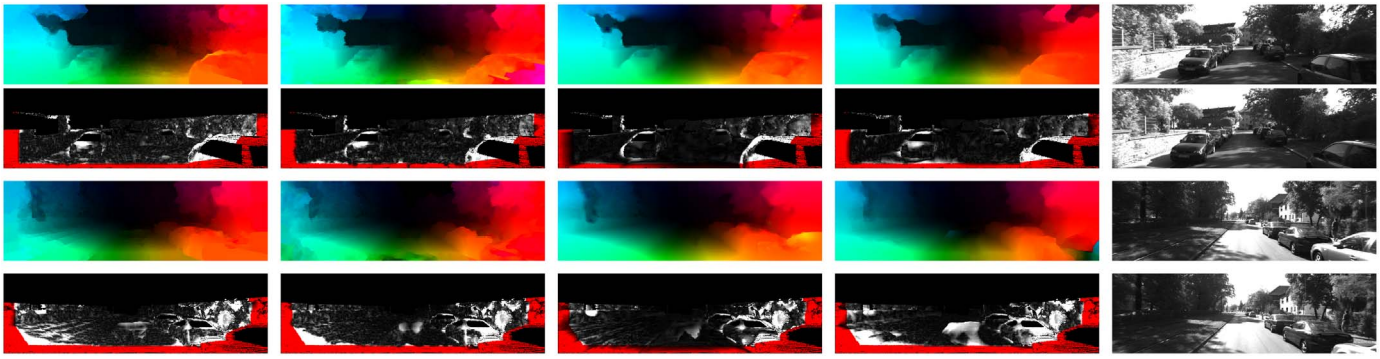

Fig. 8. Visual comparisons on the KITTI testing set. *From top to bottom:* The first and third rows are the flow maps of sequences 11 and 15. The second and fourth rows are the end-point error maps of sequences 11 and 15. *From left to right:* results of ROF-NND [43], MLDP-OF [20], NLTGV-SC [19], our NNF+EAC method, and the corresponding input images (the odd rows are frame 10 and the even rows are frame 11).

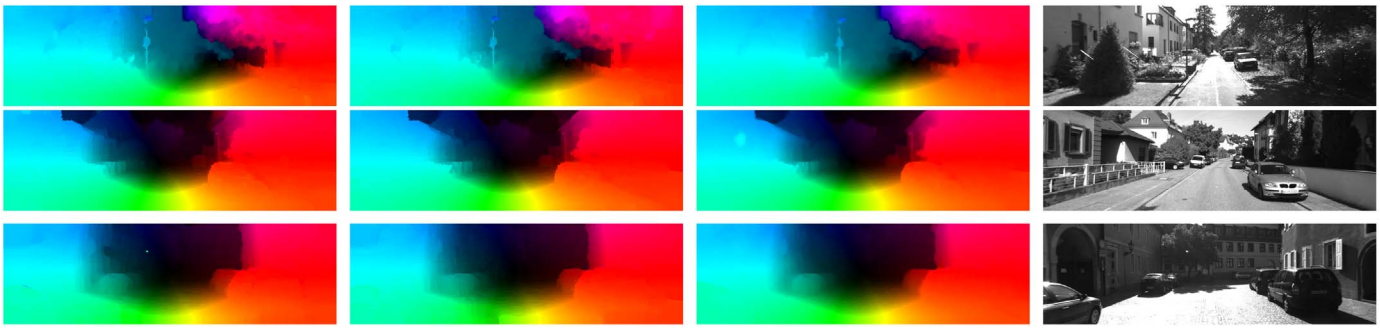

Fig. 9. Visual comparisons on the KITTI testing set. *From top to bottom:* sequences 1–3 on the testing set. *From left to right:* results of Classic+NL [36], NNF-Local [44], our NNF +EAC method, and the corresponding input images.

NNF+EAC, while they are not recognizable in the other methods. Similarly, the headstock of the car near the pole can be discerned in NNF+EAC while it is not captured in the methods of Nir et al. [30], NNF (NoRestore) and NNF+IFT. In particular, the motion boundaries of [45] are severely violated due to its deblurring operation.

6.2.3. Illumination invariance test on KITTI sequences

Fig. 8 shows the flow fields and end-point error maps of sequences 11 and 15 from KITTI. For sequence 11, the recovered flow boundaries of the car at the bottom left of our NNF-EAC are easy to discern, while the boundaries of ROF-NND [43] and MLDP-OF [20] are dim. The boundaries of the tree on the top right are mixed with the sky in the flow fields of ROF-NND and MLDP-OF, partly captured by our NNF-EAC, and clearly estimated in NLTGV-SC [19]. For sequence 15, the cars in MLDP-OF and ROF-NND are difficult to distinguish and many flow errors are produced at the bottom right. In contrast, cars are better recovered in NNF-EAC. However, many errors are computed around the smooth road. The captured flow of the NLTGV-SC is the most accurate among these four methods. Fig. 8 reflects that our NNF-EAC method can handle illumination changes, but to a limited extent.

Fig. 9 shows the visual flow fields of KITTI sequences 1–3. For sequence 1, the edges of the small triangle mark are too blurry to distinguish whether the sign is a triangle object in both the Classic+NL [36] method and the NNF-Local [44] method. Due to the proposed EAC technique, the small triangle mark is precisely recovered. Furthermore, not only the motion of the trees on the right side are accurately estimated but also edges are well preserved. For sequence 2, the window on the left side can be identified in NNF-EAC, while in Classic+NL [36] and NNF-Local [44], it is blended with the wall and cannot be discerned. For sequence 3, our method also performs best. The boundaries of the two cars on the right side in our optical flow are distinguishable. In contrast, for the other two methods, they are too blurry to recognize.

7. Conclusion

In this paper, we incorporate edge-aware constraints (EAC) into the image fidelity term (IFT) to construct a new variational model for joint optical flow computation and image restoration. By adopting an alternating optimization strategy, the optical flow and the image denoising can be solved simultaneously. The coupled IFT leads to improved accuracy of the flow estimation under noise compared to traditional flow algorithms. This is due to the IFT's ability to provide refined filtered images for flow calculation in the next iteration of minimization. Two main benefits originate from the EAC. First, the EAC imposed to get desired pixel-gradients over space and time, which causes edges to be better preserved during image filtering. Second, because the integrated EAC can be rewritten into gradient form, denoising is accurately and efficiently carried out. Compared to the classical IFT method, the most significant achievement of our EAC incorporated IFT method is that it saves more than 60% computational time.

Limitations: When γ is small, the edge preservation due to the EAC is limited, whereas large γ causes the EAC to negatively influence the image denoising. It would therefore be beneficial to find a mathematical measure to adaptively determine the value of γ for different sequences.

A second limitation is that we use a single value for α. α is related to the noise degree of the input images, and it should be reduced as the noise degree increases. Estimating α automatically would be a desirable extension of our method.

Third, denoising the input images during optimization is not always beneficial for improving the accuracy of the flow field. For example, as shown in Table 2, the results on the *RubberWhale* sequence are degraded when using the EAC-based image denoising. This is because the *RubberWhale* is clean and denoising will then remove some useful details.

Future work: There are several ways in which we can address these limitations, and further increase the practical application of our method. To modify the performance of the EAC added IFT method, research on finding the appropriate γ and α selection strategy is advisable. For example, establishing the statistical relationship between the noise degree and α is promising. Furthermore, we also hope to extend this method to other applications, like optical flow estimation in the presence of motion blur. Finally, we are looking into ways to speed up the performance by calculating some steps on GPUs.

Acknowledgments

This work was partly supported in part by the Grant (#1135616) from the National Science Foundation (NSF), the National Natural Science Foundation of China (61501198), and Wuhan Youth Science and Technology Chenguang program (2014072704011248).

References

[1] L. Xu, J. Jia, Y. Matsushita, Motion detail preserving optical flow estimation, IEEE Trans. Pattern Anal. Mach. Intell. 16 (9) (2012) 1744–1757.
[2] K. Chen, D.A. Lorenz, Image sequence interpolation based on optical flow, segmentation, and optimal control, IEEE Trans. Image Process. 21 (3) (2012) 1020–1030.
[3] C. Liu, D. Sun, A Bayesian approach to adaptive video super resolution, in: Proceedings of the Computer Vision and Pattern Recognition, 2011, pp. 209–216.
[4] A. Papazoglou, V. Ferrari, Fast object segmentation in unconstrained video, in: Proceedings of the International Conference on Computer Vision, 2013, pp. 1777–1784.
[5] P. Ji, H. Li, M. Salzmann, Y. Zhong, Robust multi-body feature tracker: a segmentation-free approach, in: Proceedings of the Computer Vision and Pattern Recognition, 2016, pp. 3843–3851.
[6] H. Wang, A. Klaser, C. Schmid, C. Liu, Dense trajectories and motion boundary descriptors for action recognition, Int. J. Comput. Vis. 103 (1) (2013) 60–79.
[7] Z. Tu, J. Cao, Y. Li, B. Li, MSR-CNN: applying motion salient region based descriptors for action recognition, in: Proceedings of the International Conference on Pattern Recognition, 2016.
[8] J. Diaz, E. Ros, S. Mota, G. Botella, A. Canas, S. Sabatini, Optical flow for cars overtaking monitor: the rear mirror blind spot problem, Ecovision (European Research Project), 2003.
[9] S. Mota, E. Ros, J. Daz, G. Botella, F. Vargas-Martin, A. Prieto, Motion driven segmentation scheme for car overtaking sequences, in: Proceedings of the International Conference on Vision in Vehicles, 2003.
[10] T. Brox, J. Malik, Large displacement optical flow: descriptor matching in variational motion estimation, IEEE Trans. Pattern Anal. Mach. Intell. 33 (3) (2011) 500–513.
[11] B. Horn, B. Schunck, Determining optical flow, Artif. Intell. 17 (1981) 185–203.
[12] Z. Tu, N. Aa, C.V. Gemeren, R.C. Veltkamp, A combined post-filtering method to improve accuracy of variational optical flow estimation, Pattern Recognit. 47 (5) (2014) 1926–1940.
[13] S. Baker, D. Scharstein, J.P. Lewis, S. Roth, M.J. Black, R. Szeliski, A database and evaluation methodology for optical flow, Int. J. Comput. Vis. 92 (1) (2011) 1–31.
[14] A. Wedel, T. Pock, C. Zach, H. Bischof, D. Cremers, An improved algorithm for TV-L1 optical flow computation, in: Dagstuhl Visual Motion Analysis Workshop, 2008.
[15] Z. Tu, R. Poppe, R.C. Veltkamp, Estimating accurate optical flow in the presence of motion blur, J. Electron. Imaging 24 (5) (2015) 053018.
[16] M.J. Black, P. Anandan, The robust estimation of multiple motions: parametric and piecewise smooth flow fields, Comput. Vis. Image Understand. 63 (1) (1996) 75–104.
[17] Z. Tu, R. Poppe, R.C. Veltkamp, Adaptive guided image filter for warping in variational optical flow, Signal Process. 127 (2016) 253–265.
[18] N. Papenberg, A. Bruhn, T. Brox, S. Didas, J. Weickert, Highly accurate optic flow computation with theoretically justified warping, Int. J. Comput. Vis. 67 (2) (2006) 141–158.
[19] R. Ranftl, K. Bredies, T. Pock, Non-local total generalized variation for optical flow estimation, in: Proceedings of the European Conference on Computer Vision, 2014, pp. 439–454.
[20] M. Mohamed, H. Rashwan, B. Mertsching, M. Garcia, D. Puig, Illumination-robust optical flow approach using local directional pattern, IEEE Trans. Circuits Syst. Video Technol. 24 (9) (2014) 1499–1508.
[21] Z. Chen, H. Jin, Z. Lin, S. Cohen, Y. Wu, Large displacement optical flow from nearest neighbor fields, in: Proceedings of the Computer Vision and Pattern Recognition, 2013, pp. 2443–2450.
[22] H. Zimmer, A. Bruhn, J. Weickert, Optic flow in harmony, Int. J. Comput. Vis. 93 (3) (2011) 368–388.
[23] D. Fortun, P. Bouthemy, C. Kervrann, Optical flow modeling and computation: a survey, Comput. Vis. Image Understand. 134 (2015) 1–21.
[24] Z. Tu, W. Xie, W. Hurst, S. Xiong, Q. Qin, Weighted root mean square approach to select the optimal smoothness parameter of the variational optical flow algorithms, Opt. Eng. 51 (3) (2012) 037202–037209.

[25] Z. Tu, R. Poppe, R.C. Veltkamp, Weighted local intensity fusion method for variational optical flow estimation, Pattern Recognit. 50 (2016) 223–232.
[26] N.P. Galatsanos, A.K. Katsaggelos, Methods for choosing the regularization parameter and the noise variance in image restoration and their relation, IEEE Trans. Image Process. 1 (1992) 322–336.
[27] L. Raket, Local smoothness for global optical flow, in: International Conference on Image Processing, 2012, pp. 1–4.
[28] D. Sun, S. Roth, M.J. Black, A quantitative analysis of current practices in optical flow estimation and the principles behind them, Int. J. Comput. Vis. 106 (2) (2014) 115–137.
[29] Z. Tu, C.V. Gemeren, R.C. Veltkamp, Improved color patch similarity measure based weighted median filter, in: Proceedings of the Asian Conference on Computer Vision, 2015, pp. 1–15.
[30] T. Nir, R. Kimel, A. Bruckstein, Variational Approach for Joint Optic-Flow Computation and Video Restoration, Technical Report CIS200503, Technion, Israel Institute of Technology, 2005.
[31] M. Hua, X. Bie, M. Zhang, W. Wang, Edge-aware gradient domain optimization framework for image filtering by local propagation, in: Proceedings of the Computer Vision and Pattern Recognition, 2014, pp. 2838–2845.
[32] P. Anandan, A computational framework and an algorithm for the measurement of visual motion, Int. J. Comput. Vis. 2 (3) (1989) 283–310.
[33] D. Sun, S. Roth, J.P. Lewis, M.J. Black, Optical flow estimation with channel constancy, in: Proceedings of the European Conference on Computer Vision, 2008, pp. 83–97.
[34] J. Xiao, H. Cheng, H. Sawhney, C. Rao, M. Isnardi, Bilateral filtering-based optical flow estimation with occlusion detection, in: Proceedings of the European Conference on Computer Vision, 2006, pp. 211–224.
[35] M. Drulea, S. Nedevschi, Motion estimation using the correlation transform, IEEE Trans. Image Process. 22 (8) (2013) 3260–3270.
[36] D. Sun, S. Roth, M.J. Black, Secrets of optical flow estimation and their principles, in: Proceedings of the Computer Vision and Pattern Recognition, 2010, pp. 2432–2439.
[37] M. Werlberger, T. Pock, M. Unger, H. Bischof, Optical flow guided TV–L1 video interpolation and restoration, in: Proceedings of the EMMCVPR, 2011, pp. 273–286.
[38] J. Wulff, M.J. Black, Modeling blurred video with layers, in: Proceedings of the European Conference on Computer Vision, 2014, pp. 236–252.
[39] L.I. Rudin, S. Osher, E. Fatemi, Nonlinear total variation based noise removal algorithms, Physica D 60 (1) (1992) 259–268.
[40] L. Xu, C. Lu, Y. Xu, J. Jia, Image smoothing via L0 gradient minimization, ACM Trans. Graph. 30 (5) (2011) 174:1–174:11.
[41] C. Tomasi, R. Manduchi, Bilateral filtering for gray and color images, in: Proceedings of the International Conference on Computer Vision, 1998, pp. 839–846.
[42] D.M. Young, Iterative Solution of Large Linear Systems, Academic Press, New York, 1971.
[43] S. Ali, C. Daul, E. Galbrun, W. Blondel, Illumination invariant optical flow using neighborhood descriptors, Comput. Vis. Image Understand. 145 (2015) 95–110.
[44] Z. Chen, H. Jin, Z. Lin, S. Cohen, Y. Wu, Large displacement optical flow with nearest neighbor field, IEEE Trans. Pattern Anal. Mach. Intell., 2014, submitted for publication.
[45] T. Portz, L. Zhang, H. Jiang, Optical flow in the presence of spatially-varying motion blur, in: Proceedings of the Computer Vision and Pattern Recognition, 2012, pp. 1752–1759.
[46] C. Vogel, S. Roth, K. Schindler, An evaluation of data costs for optical flow, in: Proceedings of the German Conference on Pattern Recognition, vol. 8142, 2013, pp. 343–353.
[47] C. Liu, W.T. Freeman, E.H. Adelson, Y. Weiss, Human assisted motion annotation, in: Proceedings of the Computer Vision and Pattern Recognition, 2008, pp. 1–8.
[48] D.J. Butler, J. Wul, G.B. Stanley, M.J. Black, A naturalistic open source movie for optical flow evaluation, in: Proceedings of the European Conference on Computer Vision, 2012, pp. 611–625.
[49] A. Blake, A. Zisserman, Visual Reconstruction, The MIT Press, Cambridge, Massachusetts, 1987.
[50] J. Barron, D. Fleet, S. Beauchemin, Performance of optical flow techniques, Int. J. Comput. Vis. 12 (1) (1994) 43–77.
[51] M. Otte, H.H. Nagel, Optical flow estimation: Advances and comparisons, in: Proceedings of the European Conference on Computer Vision, 1994, pp. 51–60.

Zhigang Tu started his MPhil PhD in image processing at the School of Electronic Information, Wuhan University, China, 2008. He received a PhD degree in communication and information system from Wuhan University, 2013. In 2015, he received a PhD degree in computer science from Utrecht University, Netherlands. He is currently a postdoctoral researcher at the School of Computing, Informatics, Decision System Engineering, Arizona State University, US. His research interests include motion estimation, action recognition, object tracking, super-resolution construction, and human-computer interaction.

Wei Xie is an associate professor at Computer School of Central China Normal University, China. His research interests include motion estimation super resolution reconstruction, image fusion and image enhancement. He received his BE degree in electronic information engineering and PhD degree in communication and information system from Wuhan University, China, in 2004 and 2010, respectively. Then, from 2010 to 2013, he served as an assistant professor at Computer School of Wuhan University, China.

Coert van Gemeren received his master's degree in cognitive artificial intelligence from the Humanities faculty of Utrecht University in 2012. After his graduate studies, he became a PhD candidate at the Science Faculty of Utrecht University. He is a computer vision researcher for the Interaction Technology group there, with a focus on the development of algorithms for interaction and mood classification in videos of groups of people.

Ronald Poppe received a PhD in computer science from the University of Twente, the Netherlands. In 2009, 2010 and 2012, he was a visiting researcher at the Delft University of Technology, Stanford University and University of Lancaster, respectively. He is currently an assistant professor at the information and computing sciences department of Utrecht University. His research interests include the analysis of human behavior from videos and other sensors, the understanding and modeling of human (communicative) behavior and the applications of both in real-life settings. In 2012 and 2013, he received the most cited paper award from the “Image and Vision Computing” journal, published by Elsevier.

Jun Cao is a senior software engineer at Manufacturing Automation group of Intel Corporation, and currently working on PhD in computer science at Arizona State University on part time base. He received MS in computer science and MS in physics from the University of Georgia in 1996 and 1997, respectively, and has been working as software engineer since then. His research interests include computer vision, industry engineering, and industrial robotics for semiconductor fabrication and testing.

Remco C. Veltkamp is full professor of multimedia at Utrecht University, Netherlands. His research interests are the analysis, recognition and retrieval of, and interaction with, music, images, and 3D objects and scenes, in particular the algorithm and experimentation aspects. He has written over 150 refereed papers in reviewed journals and conferences, and supervised 15 PhD theses. He was director of the national project GATE – Game Research for Training and Entertainment.